Available online at www.sciencedirect.com
SCIENCE DIRECT®
Comput. Methods Appl. Mech. Engrg. 192 (2003) 2299–2318
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

An efficient algebraic multigrid preconditioned conjugate gradient solver

Chihiro Iwamura [a,*], Franco S. Costa [b], Igor Sbarski [a], Alan Easton [c], Nian Li [c]

[a] *Industrial Research Institute Swinburne, Swinburne University of Technology, P.O. Box 218, Hawthorn, Vic. 3122, Australia*
[b] *Moldflow Pty Ltd., Kilsyth, Vic. 3137, Australia*
[c] *Centre for Mathematical Modelling, Swinburne University of Technology, P.O. Box 218, Hawthorn, Vic. 3122, Australia*

Received 20 September 2001; received in revised form 24 May 2002; accepted 28 May 2002

Abstract

In this paper, we present a robust and efficient algebraic multigrid preconditioned conjugate gradient solver for systems of linear equations arising from the finite element discretization of a scalar elliptic partial differential equation of second order on unstructured meshes. The algebraic multigrid (AMG) method is one of most promising methods for solving large systems of linear equations arising from unstructured meshes. The conventional AMG method usually requires an expensive setup time, particularly for three dimensional problems so that generally it is not used for small and medium size systems or low-accuracy approximations. Our solver has a quick setup phase for the AMG method and a fast iteration cycle. These allow us to apply this solver for not only large systems but also small to medium systems of linear equations and also for systems requiring low-accuracy approximations.

Keywords: Preconditioned conjugate gradient; Multigrid; AMG

1. Introduction

During numerical flow simulation of polymer injection molding, systems of linear equations with symmetric positive definite (SPD) matrices ranging in size from small to very large are solved. The finite element discretization of the fluid dynamics pressure equation on unstructured tetrahedral meshes often produces an ill-conditioned matrix. It is therefore difficult to obtain solutions of such systems of linear equations efficiently using many common iterative methods. In this paper, we use small, medium, and large size systems which are categorized by the number of unknowns being: less than 3000, 3000–20,000, and greater than 20,000, respectively.

* Corresponding author.

0045-7825/03/$ - see front matter
doi:10.1016/S0045-7825(02)00378-X

For ill-conditioned matrix problems, the preconditioned conjugate gradient (PCG) method can be efficient and robust. For relatively small size problems, common (one-level) PCG solvers, such as symmetric successive overrelaxation preconditioned conjugate gradient (SSORPCG) are reasonably efficient. However, the efficiency often deteriorates for medium to very large size problems. For these types of problems, the multigrid (multi-level) PCG method can be a fast solution method. In this method, the multigrid cycle is used as a preconditioner for the conjugate gradient (CG) method.

Our systems of linear equations arise from the finite element discretization of a scalar elliptic partial differential equation (PDE) of second order on unstructured tetrahedral meshes in polymer injection molding simulation. The geometry of our models can be very complicated and the coefficients of the differential operator can vary strongly. For this kind of problem the geometric multigrid method is not suitable because it usually requires a uniform coarsening pattern and differential operators with continuous slowly varying or constant coefficients. Furthermore, to ensure that implementation into existing commercial codes is easy, we are only interested in a black-box solver, which only requires the information contained in the given system matrix. Therefore, we have not pursued those multigrid methods which rely upon knowledge of the element formulation or nodal coordinates. For these reasons, we believe the algebraic multigrid (AMG) preconditioned CG method is the most appropriate for our purpose.

The ideal matrices for the AMG method described by Ruge and Stüben [1,2] are weakly diagonally dominant symmetric M-matrices. A matrix is a symmetric M-matrix if it is SPD and has nonpositive off-diagonal values. The publicly available AMG solver created by Ruge, Stüben and Hempel (AMG1R5) can be used as a stand alone AMG solver (AMG1R5/SA) or the AMG cycle can be used as a preconditioner for a CG solver (AMG1R5/CG). AMG1R5 is very efficient for solving systems with a symmetric M-matrix. However, the efficiency quickly deteriorates if the matrix deviates from being a symmetric M-matrix.

We have found that a simple modification to the AMG algorithm of AMG1R5 can produce a more robust and efficient solver for problems when the modified AMG cycle is used as a preconditioner for CG. The robustness is improved substantially by creating an appropriate AMG interpolation operator and strictly performing the AMG cycle in a symmetric manner to produce a SPD AMG preconditioner. We found that our systems of linear equations often contain as many as 20–35% of nonzero entries being off-diagonally positive and that they deviate strongly from a weakly diagonally dominant symmetric M-matrix. However, our approach robustly solves these systems of linear equations when this AMG preconditioner is used for CG.

The efficiency is particularly improved by reducing the number of nonzero entries of the coarse system matrices by using very small sets of interpolatory points. Firstly, we created very small sets of interpolatory points only from direct neighbors. This gives a simple, quick coarsening process and simpler coarse systems (smaller numbers of nonzero entries for the coarser system matrices). Using this approach, the small total number of nonzero entries of the coarse system matrices are mainly achieved by substantially reducing the number of nonzero entries of the first coarse system matrix and also some reduction in the second and third coarse system matrices. If the AMG1R5 coarsening strategy is used, the number of nonzero entries of the matrix of the first coarse system is often approximately the same or even larger than the number of nonzero entries of the original (finest) matrix. This large number of nonzero entries of the first coarse system contributes substantially to the total combined number of nonzero entries of all systems. We recognize that with our method there will be an increase of convergence factor, however, the benefit of simpler coarse systems greatly outweighs this disadvantage. In addition, further reduction in the total number of nonzero entries is possible without significantly increasing the convergence factor by approximating the matrix of the original system with a simpler symmetric M-matrix. The AMG cycle is applied to solve a so-called a priori preconditioning system which uses this symmetric M-matrix. This symmetric M-matrix is created from the original system matrix by simply adding any positive off-diagonal entries to the diagonal entry of the same row and replacing the positive off-diagonal entries with zero. The way the a priori preconditioning matrix is created implies a lower number of nonzero entries. Our coarsening approach can be applied to this

symmetric M-matrix system to create much lower total number of nonzero entries. The coarse systems for AMG created by these approaches are appropriate to use in preconditoning and they give three important advantages: the reduction of setup time, reduction of AMG cycle time, and reduction of memory requirement. For the sake of accuracy, it is necessary to use double (64-bits) precision variables in the CG method for most of our problems. However, it is not necessary to use double precision variables when estimating the solution of the a priori preconditioning system. Therefore, the expensive (in terms of both memory and computational work) multigrid cycle can be carried out using single (32-bits) precision variables. Hence, the efficiency is improved substantially while the memory requirement is kept small.

Tests were carried out to demonstrate that our AMG preconditioned CG method produces an efficient solution strategy when solving any size of ill-conditioned system of linear equations. In the tests, the system of linear equations for pressure for three dimensional (3D) flow simulation problems was solved for the filling phase of polymer injection molding. The solvers used to solve these systems of linear equations are

- SSORPCG: symmetric successive overrelaxation preconditioned CG, a common (one-level) PCG solver,
- AMG1R5/SA: the stand alone AMG solver of Ruge, Stüben and Hempel,
- AMG1R5/CG: the AMG solver of Ruge, Stüben and Hempel where the AMG cycle is used as a preconditioner for CG,
- SAMG: The fast commercial AMG solver (SAMG, release 20b1) which is the successor of RAMG05 [2] where RAMG05 is the successor of AMG1R5, in these tests the AMG cycle is used as a preconditioner for CG,
- AMGPCG0: our AMG preconditioned CG solver which uses the original matrix for the a priori preconditioning system and uses the simpler coarse systems for AMG,
- AMGPCG1: our AMG preconditioned CG solver which always uses a symmetric M-matrix for the a priori preconditioning system and uses the simpler coarse systems for AMG.

Throughout this paper AMGPCG is used to denote the group of AMGPCG0 and AMGPCG1, which share a common coarsening algorithm.

2. AMG preconditioner

Consider a system of linear equations

$$\mathbf{A}\mathbf{u} = \mathbf{f}, \tag{1}$$

where $\mathbf{A} \in \Re^{n\times n}$ is a SPD matrix and $\mathbf{f} \in \Re^n$ is a given column vector. The linear stationary iterative methods of first degree for solving Eq. (1) may be expressed in the form [3]

$$\mathbf{u}^{(k+1)} = (\mathbf{I} - \mathbf{Q}^{-1}\mathbf{A})\mathbf{u}^{(k)} + \mathbf{Q}^{-1}\mathbf{f} = \mathbf{G}\mathbf{u}^{(k)} + \mathbf{Q}^{-1}\mathbf{f}, \quad k = 0, 1, 2, 3, \ldots, \tag{2}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{G} \in \Re^{n\times n}$ is called the iteration matrix for the method and $\mathbf{Q} \in \Re^{n\times n}$ is a splitting matrix for the method. $\mathbf{Q}$ can be used as a preconditioner for CG if $\mathbf{Q}$ is SPD.

The PCG method obtains the solution of Eq. (1) indirectly by solving the preconditioned system [3]

$$\hat{\mathbf{A}}\hat{\mathbf{u}} = \hat{\mathbf{f}}, \tag{3}$$

where $\hat{\mathbf{A}} = \mathbf{W}^{-1}\mathbf{A}(\mathbf{W}^{-1})^{\mathrm{t}}$ is SPD, $\hat{\mathbf{u}} = \mathbf{W}^{\mathrm{t}}\mathbf{u}$, $\hat{\mathbf{f}} = \mathbf{W}^{-1}\mathbf{f}$, and $\mathbf{W}$ is such that $\mathbf{Q} = \mathbf{W}\mathbf{W}^{\mathrm{t}}$. Since the condition number of $\hat{\mathbf{A}}(\kappa(\hat{\mathbf{A}}))$ is usually very small compared to the condition number of $\mathbf{A}$ (i.e., $\kappa(\hat{\mathbf{A}}) = \kappa(\mathbf{Q}^{-1}\mathbf{A}) \ll \kappa(\mathbf{A})$), the number of necessary iterations is much smaller than solving the original problem. Note that $\mathbf{Q}$ here is not the splitting matrix for an iterative method of the original system (1), but rather the splitting matrix for an iterative method of the a priori preconditioning system

$$\mathbf{B}\mathbf{x} = \mathbf{r}, \tag{4}$$

where $\mathbf{B} \in \Re^{n\times n}$ is called the a priori preconditioning matrix and $\mathbf{r} \in \Re^n$ is the current residual of PCG. Often the PCG method uses $\mathbf{B} = \mathbf{A}$. However this is not a necessary condition. The essential aspect is that $\mathbf{Q}$ for an iterative method of a priori preconditioning system should be close to $\mathbf{A}$ in the sense of spectral equivalence so that the condition number $\kappa(\hat{\mathbf{A}})$ can be small.

Our AMGPCG solvers estimate the solution of Eq. (4) using one iteration of the multigrid V-cycle [4,5] operator $\mathbf{M}$ with zero initial estimates, where $\mathbf{M}$ corresponds to the iteration matrix of the multigrid method for Eq. (4) (i.e., equivalent to $\mathbf{G}$ in Eq. (2)). This implies that the estimated solution of Eq. (4) is the solution of the equation

$$\mathbf{Q}\mathbf{z} = \mathbf{r}, \tag{5}$$

where $\mathbf{Q}$ is the preconditioner of our AMGPCG solvers and can be expressed as

$$\mathbf{Q} = \mathbf{B}(\mathbf{I} - \mathbf{M})^{-1}. \tag{6}$$

$\mathbf{B} = \mathbf{A}$ for AMGPCG0 and $\mathbf{B}$ is a simple SPD approximation to $\mathbf{A}$ for AMGPCG1.

The multigrid method requires coarse levels, therefore we also use the expression

$$\mathbf{Q}_l = \mathbf{B}_l(\mathbf{I}_l - \mathbf{M}_l)^{-1}, \tag{7}$$

where subscript l indicates level. $l = 1$ corresponds to the finest level and $l = 2$ to the first coarse level and so on (throughout this paper, we usually omit l for the finest level or if there is no need to distinguish the levels). Here $\mathbf{B}_l \in \Re^{n_l\times n_l}$ is the matrix of the system at level l. Coarser level matrices $\mathbf{B}_{l+1} \in \Re^{n_{l+1}\times n_{l+1}}$, where $n_{l+1} < n_l$, are created by the Galerkin formulation [2,6],

$$\mathbf{B}_{l+1} = \mathbf{I}_l^{l+1}\mathbf{B}_l\mathbf{I}_{l+1}^l, \tag{8}$$

where $\mathbf{I}_{l+1}^l : \Re^{n_{l+1}} \to \Re^{n_l}$ denotes the coarse-to-fine interpolation operator and $\mathbf{I}_l^{l+1} : \Re^{n_l} \to \Re^{n_{l+1}}$ denotes the fine-to-coarse restriction operator. $\mathbf{I}_l$ is the identity matrix at level l. $\mathbf{M}_l$ is one iteration of the multigrid V-cycle operator for level l which is denoted by

$$\mathbf{M}_l = \mathbf{S}_l^{(\mathrm{post})}\mathbf{T}_{l,l+1}\mathbf{S}_l^{(\mathrm{pre})}, \tag{9}$$

where $\mathbf{S}_l^{(\mathrm{pre})}$, $\mathbf{S}_l^{(\mathrm{post})}$ are pre- and post-smoothing operators, respectively, and $\mathbf{T}_{l,l+1}$ is a coarse grid correction operator. In AMGPCG, we use one iteration of forward Gauss–Seidel for pre-smoothing and backward Gauss–Seidel for post-smoothing. These pre- and post-smoothing operators can be defined as

$$\begin{aligned} \mathbf{S}_l^{(\mathrm{pre})} &= \mathbf{I}_l - \mathbf{L}_l^{-1}\mathbf{B}_l, \\ \mathbf{S}_l^{(\mathrm{post})} &= \mathbf{I}_l - (\mathbf{L}_l^{-1})^{\mathrm{t}}\mathbf{B}_l, \end{aligned} \tag{10}$$

where $\mathbf{L}_l$ is the lower triangular part of matrix $\mathbf{B}_l$ including the diagonal entries. The coarse grid correction operator $\mathbf{T}_{l,l+1}$ can be expressed as

$$\mathbf{T}_{l,l+1} = \begin{cases} \mathbf{I}_l - \mathbf{I}_{l+1}^l(\mathbf{B}_{l+1})^{-1}\mathbf{I}_l^{l+1}\mathbf{B}_l & \text{if } l = l_{\max} - 1, \\ \mathbf{I}_l - \mathbf{I}_{l+1}^l(\mathbf{I}_{l+1} - \mathbf{M}_{l+1})(\mathbf{B}_{l+1})^{-1}\mathbf{I}_l^{l+1}\mathbf{B}_l & \text{if } 1 \leqslant l < l_{\max} - 1, \end{cases} \tag{11}$$

where $l_{\max}$ indicates the coarsest level.

This $\mathbf{Q}$ needs to be SPD to use it as a preconditioner for CG and it is strongly desirable that $\mathbf{Q}$ is close to $\mathbf{A}$ and easy to invert for the efficiency of CG. We consider the following four assumptions:

(i) The a priori preconditioning matrix $\mathbf{B}$ is SPD and satisfies the spectral equivalence inequality to matrix $\mathbf{A}$, i.e.,

$$\alpha_{\mathbf{B}}(\mathbf{Bu},\mathbf{u}) \leqslant (\mathbf{Au},\mathbf{u}) \leqslant \beta_{\mathbf{B}}(\mathbf{Bu},\mathbf{u}) \quad \forall \mathbf{u} \in \Re^n \tag{12}$$

with positive spectral constants $\alpha_{\mathbf{B}}$ and $\beta_{\mathbf{B}}$ being as close as possible to each other. Here $(.,.)$ denotes the usual Euclidean inner product.

(ii) The restriction operator $\mathbf{I}_l^{l+1}$ is the transposed matrix to the interpolation operator $\mathbf{I}_{l+1}^l$ (i.e., $\mathbf{I}_l^{l+1} = (\mathbf{I}_{l+1}^l)^{\mathrm{t}}$ for all $l = 1,\ldots,l_{\max}-1$).

(iii) The corresponding coarse matrices of $\mathbf{B}$ are SPD (i.e., $\mathbf{B}_l$ is SPD for all $l = 2,\ldots,l_{\max}$).

(iv) The post-smoothing operator $\mathbf{S}_l^{(\mathrm{post})}$ is adjoint to the pre-smoothing operator $\mathbf{S}_l^{(\mathrm{pre})}$ in the $\mathbf{B}_l$ energy inner product, i.e.,

$$(\mathbf{S}_l^{(\mathrm{pre})}\mathbf{u}_l,\mathbf{v}_l)_{\mathbf{B}_l} = (\mathbf{u}_l,\mathbf{S}_l^{(\mathrm{post})}\mathbf{v}_l)_{\mathbf{B}_l} \quad \forall \mathbf{u}_l,\mathbf{v}_l \in \Re^{n_l} \tag{13}$$

for all $l = 1,\ldots,l_{\max}-1$, where $(\mathbf{u}_l,\mathbf{v}_l)_{\mathbf{B}_l} = (\mathbf{B}_l\mathbf{u}_l,\mathbf{v}_l)$ for $\mathbf{u}_l,\mathbf{v}_l \in \Re^{n_l}$.

If the above assumptions (i)–(iv) are fulfilled, and furthermore, if the matrix norm of the multigrid operator $\|\mathbf{M}\|_*$ which corresponds to some vector norm $\|\cdot\|_*$ is less than one, i.e.,

$$\|\mathbf{M}\|_* = \sup_{\substack{\mathbf{v}\in\Re^n,\\ \mathbf{v}\neq\mathbf{0}}} (\|\mathbf{Mv}\|_*/\|\mathbf{v}\|_*) \leqslant \eta = \text{constant} < 1, \tag{14}$$

then Jung et al. [6] have proved that this preconditioner $\mathbf{Q}$ is SPD and that $\mathbf{Q}$ satisfies the spectral equivalence inequality to matrix $\mathbf{A}$, i.e.,

$$\alpha_{\mathbf{Q}}(\mathbf{Qu},\mathbf{u}) \leqslant (\mathbf{Au},\mathbf{u}) \leqslant \beta_{\mathbf{Q}}(\mathbf{Qu},\mathbf{u}) \quad \forall \mathbf{u} \in \Re^n, \tag{15}$$

where $\alpha_{\mathbf{Q}} = \alpha_{\mathbf{B}}(1-\eta)$ and $\beta_{\mathbf{Q}} = \beta_{\mathbf{B}}(1+\eta)$. Hence the use of this $\mathbf{Q}$ as a preconditioner for CG would be legitimate and $\mathbf{Q}$ will be an efficient preconditioner.

In AMGPCG, we construct the coarse matrices using Eq. (8), where $\mathbf{I}_l^{l+1} = (\mathbf{I}_{l+1}^l)^{\mathrm{t}}$ and $\mathbf{I}_{l+1}^l$ has full rank. We use the pre-smoothing and post-smoothing defined in Eq. (10), and use the coarse grid correction operator defined in Eq. (11). With these components, the multigrid operator defined in Eq. (9) becomes self-adjoint in the $\mathbf{B}_l$ energy inner product and satisfies $\|\mathbf{M}_l\|_{\mathbf{B}_l} < 1$ for all $l = 1,\ldots,l_{\max}-1$. The proof of $\|\mathbf{M}_l\|_{\mathbf{B}_l} < 1$ is shown in the next section. Therefore $\mathbf{Q}$ becomes SPD. $\mathbf{Q}$ is symmetric because $\mathbf{M}_l$ is self-adjoint in the $\mathbf{B}_l$ energy inner product. $\mathbf{Q}$ is well defined and positive definite because $\|\mathbf{M}_l\|_{\mathbf{B}_l} < 1$ (see [6]). For AMGPCG1, we will show in Section 4 the existence of two positive values $\alpha_{\mathbf{B}}$ and $\beta_{\mathbf{B}}$ which satisfy inequality (12). Hence the preconditioners of AMGPCG are legitimate preconditioners to use for CG.

3. Robustness for multigrid V-cycle operator

In this section we do not say anything about the efficiency for the multigrid V-cycle operator, however we show that $\|\mathbf{M}\|_{\mathbf{B}} < 1$ so that robustness is assured and the multigrid preconditioner is legitimate.

Gauss–Seidel smoothing operators $\mathbf{S}$ (both $\mathbf{S}^{(\mathrm{pre})}$ and $\mathbf{S}^{(\mathrm{post})}$) will have the following relation for any SPD matrix under reasonable assumptions (see [2, p. 30]):

$$\|\mathbf{S}_l\mathbf{e}_l\|_{\mathbf{B}_l}^2 \leqslant \|\mathbf{e}_l\|_{\mathbf{B}_l}^2 - \sigma\|\mathbf{e}_l\|_{\mathbf{B}_l\mathbf{D}_l^{-1}\mathbf{B}_l}^2, \tag{16}$$

where $\|*\|_{\mathbf{B}_l}$ is the energy norm with respect to SPD matrix $\mathbf{B}_l$, $\|*\|_{\mathbf{B}_l\mathbf{D}_l^{-1}\mathbf{B}_l}$ is a norm with respect to SPD matrix $\mathbf{B}_l\mathbf{D}_l^{-1}\mathbf{B}_l$ (i.e., $\|\mathbf{e}_l\|_{\mathbf{B}_l\mathbf{D}_l^{-1}\mathbf{B}_l}^2 = (\mathbf{B}_l\mathbf{D}_l^{-1}\mathbf{B}_l\mathbf{e}_l,\mathbf{e}_l)$), here $\mathbf{D}_l^{-1}$ is the inverse of the diagonal matrix of $\mathbf{B}_l$, $\mathbf{e}_l$ is the error vector at level l, and $\sigma > 0$. We have

$$\|\mathbf{e}_l\|_{\mathbf{B}_l\mathbf{D}_l^{-1}\mathbf{B}_l}^2 \geqslant \xi\|\mathbf{e}_l\|_{\mathbf{B}_l}^2, \tag{17}$$

where $\xi = \lambda_{\min}(\mathbf{D}_l^{-1})\lambda_{\min}(\mathbf{B}_l) > 0$. $\lambda_{\min}(\mathbf{D}_l^{-1})$ and $\lambda_{\min}(\mathbf{B}_l)$ are the smallest eigenvalues of $\mathbf{D}_l^{-1}$ and $\mathbf{B}_l$, respectively. We obtain the following relation by substituting inequality (17) into inequality (16)

$$\|\mathbf{S}_l\mathbf{e}_l\|_{\mathbf{B}_l}^2 \leqslant \gamma\|\mathbf{e}_l\|_{\mathbf{B}_l}^2, \tag{18}$$

where $\gamma = 1 - \sigma\xi$ and $0 < \gamma < 1$. Therefore we obtain

$$\|\mathbf{S}_l\mathbf{e}_l\|_{\mathbf{B}_l} \leqslant \gamma^{0.5}\|\mathbf{e}_l\|_{\mathbf{B}_l}, \tag{19}$$

for all l $(1 \leqslant l \leqslant l_{\max} - 1)$.

For the case of the coarse grid correction operator $\mathbf{T}_{l,l+1}$, we first consider the case $l = l_{\max} - 1$. $\mathbf{e}_l$ can be expressed using $\mathbf{T}_{l,l+1}$ and $\mathbf{I}_{l+1}^l$

$$\mathbf{e}_l = \mathbf{T}_{l,l+1}\mathbf{e}_l + (\mathbf{I}_l - \mathbf{T}_{l,l+1})\mathbf{e}_l = \mathbf{T}_{l,l+1}\mathbf{e}_l + \mathbf{I}_{l+1}^l\mathbf{e}_{l+1}, \tag{20}$$

where $\mathbf{e}_{l+1} = (\mathbf{B}_{l+1})^{-1}\mathbf{I}_l^{l+1}\mathbf{B}_l\mathbf{e}_l$. Since $(\mathbf{T}_{l,l+1}v_l, \mathbf{I}_{l+1}^l v_{l+1})_{\mathbf{B}_l} = 0$ for all v_l, v_{l+1},

$$\|\mathbf{e}_l\|_{\mathbf{B}_l}^2 = \|\mathbf{T}_{l,l+1}\mathbf{e}_l\|_{\mathbf{B}_l}^2 + \|\mathbf{I}_{l+1}^l\mathbf{e}_{l+1}\|_{\mathbf{B}_l}^2. \tag{21}$$

Then from Eq. (21)

$$\|\mathbf{T}_{l,l+1}\mathbf{e}_l\|_{\mathbf{B}_l} < \|\mathbf{e}_l\|_{\mathbf{B}_l}. \tag{22}$$

Using inequalities (19) and (22), we get

$$\|\mathbf{M}_l\mathbf{e}_l\|_{\mathbf{B}_l} = \|\mathbf{S}_l^{(\mathrm{post})}\mathbf{T}_{l,l+1}\mathbf{S}_l^{(\mathrm{pre})}\mathbf{e}_l\|_{\mathbf{B}_l} < \gamma\|\mathbf{e}_l\|_{\mathbf{B}_l}. \tag{23}$$

Therefore

$$\|\mathbf{M}_l\|_{\mathbf{B}_l} = \sup_{\mathbf{e}_l \neq 0} \frac{\|\mathbf{M}_l\mathbf{e}_l\|_{\mathbf{B}_l}}{\|\mathbf{e}_l\|_{\mathbf{B}_l}} \leqslant \gamma < 1. \tag{24}$$

When $l = l_{\max} - 2$, we have

$$\mathbf{T}_{l,l+1}\mathbf{e}_l = \mathbf{e}_l - \mathbf{I}_{l+1}^l(\mathbf{I}_{l+1} - \mathbf{M}_{l+1})(\mathbf{B}_{l+1})^{-1}\mathbf{I}_l^{l+1}\mathbf{B}_l\mathbf{e}_l. \tag{25}$$

We rewrite Eq. (25) using $\overline{\mathbf{T}}_{l,l+1}$.

$$\mathbf{T}_{l,l+1}\mathbf{e}_l = \{\overline{\mathbf{T}}_{l,l+1}\mathbf{e}_l + (\mathbf{I}_l - \overline{\mathbf{T}}_{l,l+1})\mathbf{e}_l\} - \mathbf{I}_{l+1}^l(\mathbf{e}_{l+1} - \mathbf{M}_{l+1}\mathbf{e}_{l+1}) = \overline{\mathbf{T}}_{l,l+1}\mathbf{e}_l + \mathbf{I}_{l+1}^l\mathbf{M}_{l+1}\mathbf{e}_{l+1}, \tag{26}$$

where $\overline{\mathbf{T}}_{l,l+1} = \mathbf{I}_l - \mathbf{I}_{l+1}^l(\mathbf{B}_{l+1})^{-1}\mathbf{I}_l^{l+1}\mathbf{B}_l$. Since we have $(\overline{\mathbf{T}}_{l,l+1}v_l, \mathbf{I}_{l+1}^l v_{l+1})_{\mathbf{B}_l} = 0$ for all v_l, v_{l+1} and $\|\mathbf{M}_{l+1}\mathbf{e}_{l+1}\|_{\mathbf{B}_{l+1}} < \|\mathbf{e}_{l+1}\|_{\mathbf{B}_{l+1}}$ from (23),

$$\|\mathbf{T}_{l,l+1}\mathbf{e}_l\|_{\mathbf{B}_l}^2 = \|\overline{\mathbf{T}}_{l,l+1}\mathbf{e}_l\|_{\mathbf{B}_l}^2 + \|\mathbf{I}_{l+1}^l\mathbf{M}_{l+1}\mathbf{e}_{l+1}\|_{\mathbf{B}_l}^2 < \|\overline{\mathbf{T}}_{l,l+1}\mathbf{e}_l\|_{\mathbf{B}_l}^2 + \|\mathbf{I}_{l+1}^l\mathbf{e}_{l+1}\|_{\mathbf{B}_l}^2 = \|\mathbf{e}_l\|_{\mathbf{B}_l}^2. \tag{27}$$

Therefore from (27), we again obtain the inequality (22) for $l = l_{\max} - 2$. Because our smoothing operators and coarse grid correction operator satisfy inequalities (19) and (22) respectively, our multigrid operator $\mathbf{M}_l$ satisfies $\|\mathbf{M}_l\mathbf{e}_l\|_{\mathbf{B}_l} < \gamma\|\mathbf{e}_l\|_{\mathbf{B}_l}$ and hence $\|\mathbf{M}_l\|_{\mathbf{B}_l} < 1$ for $l = l_{\max} - 2$. Similar discussion can prove $\|\mathbf{M}_l\|_{\mathbf{B}_l} < 1$ for $1 \leqslant l < l_{\max} - 2$. Therefore our multigrid operator satisfies $\|\mathbf{M}_l\|_{\mathbf{B}_l} < 1$ for all l $(1 \leqslant l \leqslant l_{\max} - 1)$.

4. A priori preconditioning matrix of AMGPCG1

The clear advantage of replacing the original matrix $\mathbf{A}$ with a simpler approximation $\mathbf{B}$ is the reduction of memory requirement and computational work necessary for both the setup and the AMG cycle.

However, the disadvantage is that the ratio of the two constants $\alpha_{\mathbf{B}}$ and $\beta_{\mathbf{B}}$ of inequality (12) $\beta_{\mathbf{B}}/\alpha_{\mathbf{B}}$ becomes larger so that the ratio of $\alpha_{\mathbf{Q}}$ and $\beta_{\mathbf{Q}}$ of inequality (15) $\beta_{\mathbf{Q}}/\alpha_{\mathbf{Q}}$ may become larger. This disadvantage is minimized by approximating the original matrix $\mathbf{A}$ with a symmetric M-matrix which is close to the original matrix $\mathbf{A}$. This is because our interpolation operator $\mathbf{I}_{l+1}^{l}$ can be more accurately defined if the matrix is a symmetric M-matrix. This implies a more accurate coarse grid correction operator $\mathbf{T}_{l,l+1}$ so that we can have a much smaller value of $\|\mathbf{M}\|_*$ for this kind of matrix. The details of defining $\mathbf{I}_{l+1}^{l}$ are discussed in Section 5.2 and estimations of $\|\mathbf{M}\|_*$ are shown in Section 6 for problems arising in polymer injection molding flow simulation.

The approximation of matrix $\mathbf{A}$ is created by

$$\begin{aligned} &b_{ii} = a_{ii} + \sum_{j \in N_i^+} a_{ij}, \quad N_i^+ = \{j : j \neq i, a_{ij} > 0\}, \\ &b_{ij} = 0 \quad \text{if } a_{ij} \geqslant 0,\ i \neq j, \\ &b_{ij} = a_{ij} \quad \text{if } a_{ij} < 0,\ i \neq j, \end{aligned} \tag{28}$$

where a_{ij} and b_{ij} are entries of matrices $\mathbf{A}$ and $\mathbf{B}$ respectively. With this approximation, matrix $\mathbf{B}$ becomes a symmetric M-matrix [7] which is approximately weakly diagonally dominant because most of the rows of matrix $\mathbf{A}$ have row summation approximately zero for our problems. This satisfies the first part of assumption (i) that the a priori preconditioning matrix is SPD. Furthermore, all eigenvalues of $\mathbf{B}^{-1}\mathbf{A}$ are in the range $(0, 1]$ [7]. Hence $\kappa(\mathbf{B}^{-1}\mathbf{A})$ is bounded by

$$\kappa(\mathbf{B}^{-1}\mathbf{A}) \leqslant 1/\lambda_{\min}(\mathbf{B}^{-1}\mathbf{A}), \tag{29}$$

where $\lambda_{\min}(\mathbf{B}^{-1}\mathbf{A})$ is smallest eigenvalue of $\mathbf{B}^{-1}\mathbf{A}$. Inequality (29) guaranties the existence of two positive values $\alpha_{\mathbf{B}}$ and $\beta_{\mathbf{B}}$ which satisfies inequality (12). Our experiments also confirmed that the eigenvalues of $\mathbf{B}^{-1}\mathbf{A}$ are in the range $(0, 1]$ and $\lambda_{\min}(\mathbf{B}^{-1}\mathbf{A})$ for our problems are usually between 0.1 to 0.5 and the maximum eigenvalue of $\mathbf{B}^{-1}\mathbf{A}$ is 1.0. Therefore, the second part of assumption (i) is also satisfied with this approach.

5. Details of the coarsening process for AMGPCG

In the AMG method, algebraically smooth errors which cannot be eliminated quickly by a smoother (standard iterative method such as Gauss–Seidel) are eliminated efficiently by using appropriately adjusted coarse systems. However, the AMG method solver has some disadvantages. These include the expensive cost of setting up these coarse systems and the large memory requirement compared to a common (one-level) iterative solver such as SSORPCG. To overcome these disadvantages we use appropriate coarse systems which have lower numbers of nonzero entries in their matrices. In AMGPCG, these appropriate coarse systems are achieved by substantially reducing the number of nonzero entries of the first coarse system matrix and also reducing somewhat the second and third coarse system matrices as outlined in the following sections.

To describe the coarsening process of AMGPCG, from fine level l to coarse level $l+1$, we rewrite Eq. (4) as

$$\mathbf{B}_l \mathbf{x}_l = \mathbf{r}_l \quad \text{or} \quad \sum_{j \in \Omega_l} b_{ij,l} x_{j,l} = r_{i,l}, \quad i \in \Omega_l, \tag{30}$$

where Ω_l is the index set $\{1, 2, 3, \ldots, n_l\}$. In order to derive the coarser level system we need to split Ω_l into disjoint subsets

$$\Omega_l = C_l \cup F_l, \tag{31}$$

where C_l contains the points also represented in the coarser level (C-points, i.e., $\Omega_{l+1} = C_l$) and F_l is the complementary set (F-points). After splitting Ω_l into C_l and F_l, the coarse level AMG system

$$\mathbf{B}_{l+1}\mathbf{x}_{l+1} = \mathbf{r}_{l+1} \quad \text{or} \quad \sum_{j\in\Omega_{l+1}} b_{ij,l+1}x_{j,l+1} = r_{i,l+1}, \quad i \in \Omega_{l+1}, \tag{32}$$

is constructed using a Galerkin formulation. The Galerkin operator, $\mathbf{B}_{l+1}$, is defined in Eq. (8). $\mathbf{r}_{l+1}$ and $\mathbf{x}_{l+1}$ actually correspond to residual and correction (error) of Eq. (30), respectively. Hence, Eq. (32) can be expressed as

$$\mathbf{I}_l^{l+1}\mathbf{B}_l\mathbf{I}_{l+1}^l\mathbf{e}_{l+1} = \mathbf{I}_l^{l+1}(\mathbf{r}_l - \mathbf{B}_l\mathbf{v}_l), \tag{33}$$

where $\mathbf{v}_l$ is the estimated solution of Eq. (30) after applying the pre-smoothing operator. Since we have $\mathbf{I}_l^{l+1} = (\mathbf{I}_{l+1}^l)^{\mathrm{t}}$, defining the interpolation operator $\mathbf{I}_{l+1}^l$ is the one of the main tasks of the coarsening process.

5.1. Sets of connections between grid points

Before defining the interpolation operator, we need to introduce sets which characterize the connections between grid points. The first set is the direct neighborhood of a point i defined as

$$N_i = \{j \in \Omega : j \neq i,\ b_{ij} \neq 0\}, \tag{34}$$

where Ω is the index set. Next we say that i is strongly negatively connected to another point j if $-b_{ij} \geqslant \varepsilon_{\mathrm{str}} \max|b_{ik}^-|$ with $0 < \varepsilon_{\mathrm{str}} < 1$, where $b_{ik}^- = b_{ik}$ if $b_{ik} < 0$ and $b_{ik}^- = 0$ if $b_{ik} \geqslant 0$, and $\varepsilon_{\mathrm{str}}$ is a constant which defines strong negative connections. Then we denote a set of all strong negative connections of point i by S_i

$$S_i = \{j \in N_i : -b_{ij} \geqslant \varepsilon_{\mathrm{str}} \max|b_{ik}^-| \quad \text{with fixed } 0 < \varepsilon_{\mathrm{str}} < 1\}. \tag{35}$$

Also, we denote the set of points strongly negatively connected to i by S_i^{t}

$$S_i^{\mathrm{t}} = \{j \in \Omega : i \in S_j\}. \tag{36}$$

In AMG1R5, $\varepsilon_{\mathrm{str}}$ is set to 0.25 as a default value. In our AMGPCG method for the systems arising from 3D unstructured meshes, we set $\varepsilon_{\mathrm{str}}$ to a very high value (i.e., $\varepsilon_{\mathrm{str}} = 0.98$) for the finest and first coarse level, 0.75 for the second coarse level, and 0.25 for the remaining coarser levels ($l = 4, \ldots, l_{\max} - 1$). The reason will be discussed below. Finally, for any set P, $|P|$ denotes the number of elements in the set P.

5.2. Defining of interpolation weight

We assume that Ω is separated into disjoint subsets C and F. Then we define the interpolation weight so that the interpolation and the restriction operators can be defined.

The ith component of error $\mathbf{e}$ at fine level l is given by

$$e_{i,l} = (\mathbf{I}_{l+1}^l\mathbf{e}_{l+1})_i = \begin{cases} e_{i,l+1} & \text{if } i \in C_l, \\ \sum_{k\in C_{i,l}} w_{ik}e_{k,l+1} & \text{if } i \in F_l, \end{cases} \tag{37}$$

where C_i is called a set of interpolatory points and is defined below. Our objective is to define the interpolation weight w_{ik} of Eq. (37).

The error $\mathbf{e}$ is considered to be algebraically smooth if $\|\mathbf{Se}\|_{\mathbf{B}} \approx \|\mathbf{e}\|_{\mathbf{B}}$, where $\mathbf{S}$ is a smoothing operator and $\|\mathbf{e}\|_{\mathbf{B}} = (\mathbf{Be}, \mathbf{e})^{1/2}$. When the error is algebraically smooth the following relation [1,2,4] holds:

$$b_{ii}e_i + \sum_{j\in N_i} b_{ij}e_j \approx 0. \tag{38}$$

The algebraically smooth error changes slowly in the direction of strong negative connections [1,2,4]. Therefore for each $i \in F$, points in N_i are usually split into three categories for all levels, except the coarsest:

- the set of C-points with which point i has strong negative connections defined as $C_i = S_i \cap C$,
- the set of F-points with which point i has strong negative connections defined as $D_i^{\mathrm{S}} = S_i - C_i$, and
- the set of points with which point i has weak connections including positive connections, which may contain both C- and F-points, defined as $D_i^{\mathrm{W}} = N_i - S_i$.

The set D_i^{W} contains both negative and positive connections. The algebraically smooth error usually does not change slowly in the direction of positive connections [2]. Hence we split D_i^{W} into the following two disjoint sets

$$D_i^{\mathrm{W}-} = \{j \in D_i^{\mathrm{W}} : b_{ij} < 0\}, \quad D_i^{\mathrm{W}+} = \{j \in D_i^{\mathrm{W}} : b_{ij} \geqslant 0\}, \tag{39}$$

in order to treat positive connections differently. Therefore, we rewrite (38) as

$$b_{ii}e_i + \sum_{j\in C_i} b_{ij}e_j + \sum_{j\in D_i^{\mathrm{W}-}} b_{ij}e_j + \sum_{j\in D_i^{\mathrm{W}+}} b_{ij}e_j + \sum_{j\in D_i^{\mathrm{S}}} b_{ij}e_j \approx 0. \tag{40}$$

To establish the interpolation weight w_{ik} of Eq. (37), it is necessary to estimate e_j $(j \notin C_i)$ of (40) by using either e_i or e_k $(k \in C_i)$. Our aim is to construct this w_{ik} as efficiently as possible while maintaining the accuracy of w_{ik} since the efficient construction of w_{ik} allows shorter setup time and an accurate w_{ik} produces a smaller convergence factor.

Firstly, we approximate e_j $(j \in D_i^{\mathrm{W}-})$ with e_i. Because any estimation error caused by this assignment is relatively insignificant if $|b_{ij}|$ is small, and e_j and e_i are close to each other if $|b_{ij}|$ is large. This estimation of e_j $(j \in D_i^{\mathrm{W}-})$ is very quick and reasonably accurate.

For a positive connection, it is also usually sufficient to estimate e_j $(j \in D_i^{\mathrm{W}+})$ by e_i since the positive connections are generally small. However, the accuracy of estimation for this e_j can be improved by using e_k $(k \in C_i)$ if there are strong negative paths from point i to j and from j to i, where strong negative paths mean that a path between points i and j through points $k \in C_i$ with relatively large negative connections exists. When there are strong negative paths from points i to j, the algebraically smooth error may still change slowly between these points. Further, if there are strong negative paths in both directions between points i and j, we can have more confidence in the assumption that the algebraically smooth error changes slowly between these points. Therefore we approximate e_j by

$$e_j \approx \sum_{k\in C_i} b_{jk}^{\mathrm{S}-} e_k \Bigg/ \sum_{k\in C_i} b_{jk}^{\mathrm{S}-}, \tag{41}$$

where $b_{jk}^{\mathrm{S}-} = b_{jk}$ if $k \in C_i^{j+} \subset C_i$ else $b_{jk}^{\mathrm{S}-} = 0$. Set C_i^{j+} is defined as

$$\begin{aligned} C_i^{j+} = \{k \in C_i : &-b_{ik} > \varepsilon_0 b_{ii}, -b_{kj} > \varepsilon_0 b_{kk}, -b_{jk} > \varepsilon_1 b_{jj}, -b_{ki} > \varepsilon_1 b_{kk}, b_{ij} < \varepsilon_2 |b_{ik}^-|, \\ &\text{and} \quad b_{ij} < \varepsilon_2 |b_{kj}^-| \quad \text{for } b_{ij} > 0 \quad \text{with } 0 < \varepsilon_0, \varepsilon_1, \varepsilon_2 < 1\}. \end{aligned} \tag{42}$$

The first and second inequalities of (42) establish strong negative paths from point i to j, and the third and fourth inequalities of (42) establish strong negative paths from point j to i. In addition, we require a relatively small b_{ij} compare to $|b_{ik}^-|$, $|b_{kj}^-|$. This condition also implies a relatively small b_{ij} compared to b_{ii} and b_{jj}, since the diagonal entry is usually largest in the row for our matrices. A small b_{ij} is necessary

because the algebraically smooth error tends to oscillate along large positive connections [2]. Our experience has shown that ε_0, ε_1 and ε_2 can be constants for each matrix, however the appropriate values of ε_0, ε_1 and ε_2 depend on matrices. For example, if the matrix contains relatively large positive off-diagonal entries the values ε_0 and ε_1 need to be large to ensure very strong two way paths between points i and j. On the other hand, ε_2 is insensitive to the condition of the matrices. The algorithm to estimate e_j using (41) is complicated compared to simply estimating it by e_i and the improvement obtained can be negated by increase of computational work necessary. Therefore, we usually apply this estimation to matrices of coarse systems which have relatively small positive off-diagonal entries, so that the only small increase of computational work is necessary and some improvement is gained.

In the case of e_j $(j \in D_i^{\mathrm{S}})$, approximation of e_j with e_i can be accurate, because the algebraically smooth error changes slowly in the direction of strong negative connections, however our experience has shown that it is better to estimate e_j by averaging negative connections rather than estimate it from a single point, i. Hence we estimate error e_j $(j \in D_i^{\mathrm{S}})$ by

$$e_j \approx \sum_{k \in C_i} b_{jk}^{-} e_k \Bigg/ \sum_{k \in C_i} b_{jk}^{-}. \tag{43}$$

Therefore (40) becomes

$$\begin{aligned} \left(b_{ii} + \sum_{j \in D_i^{\mathrm{W}-}} b_{ij} \right) e_i \approx & - \sum_{j \in C_i} b_{ij} e_j - \sum_{j \in D_i^{\mathrm{W}+}} b_{ij} \left(\sum_{k \in C_i} b_{jk}^{\mathrm{S}-} e_k \Bigg/ \sum_{l \in C_i} b_{jl}^{\mathrm{S}-} \right) \\ & - \sum_{j \in D_i^{\mathrm{S}}} b_{ij} \left(\sum_{k \in C_i} b_{jk}^{-} e_k \Bigg/ \sum_{l \in C_i} b_{jl}^{-} \right). \end{aligned} \tag{44}$$

From Eq. (37) and (44) the interpolation weight becomes

$$w_{ik} = - \left\{ b_{ik} + \sum_{j \in D_i^{\mathrm{W}+}} \left(b_{ij} b_{jk}^{\mathrm{S}-} \Bigg/ \sum_{l \in C_i} b_{jl}^{\mathrm{S}-} \right) + \sum_{j \in D_i^{\mathrm{S}}} \left(b_{ij} b_{jk}^{-} \Bigg/ \sum_{l \in C_i} b_{jl}^{-} \right) \right\} \Bigg/ \left(b_{ii} + \sum_{j \in D_i^{\mathrm{W}-}} b_{ij} \right). \tag{45}$$

When there are no two way strong negative paths between points i and $j \in D_i^{\mathrm{W}+}$ (i.e., $C_i^{j+} = \emptyset$), we modify the interpolation weight. We can approximate e_j with e_i because matrices arising from the scalar elliptic PDE in polymer injection molding flow simulation have generally small positive off-diagonal entries compared to b_{ii}. If we have relatively large positive off-diagonal entries, the accuracy of w_{ik} may deteriorate with this approximation. However, this approximation will avoid w_{ik} becoming negative which can cause poor estimation of e_i. In the case of $j \in D_i^{\mathrm{S}}$, if the negative connection to C_i is considered to be weak, j or i is put into set C (see next section for details). This is done because an accurate estimate of e_j is required, since it has a strong influence on e_i.

In theory, this w_{ik} can be negative. However, our experience has shown that w_{ik} does not become negative at all and so does not influence the accuracy of interpolation significantly. If we had estimated e_j $(j \in D_i^{\mathrm{W}-})$ using (43), w_{ik} could almost certainly have been kept positive, because b_{ij} $(j \in D_i^{\mathrm{W}-})$ is distributed to C_i for calculating the interpolation, and the row summation of most rows of our matrices is approximately zero. However, compared to simply adding b_{ij} $(j \in D_i^{\mathrm{W}-})$ to b_{ii}, this slows down the setup phase (approximately 20–30% extra setup time for our problems) while the improvement in accuracy with this w_{ik} is marginal. Therefore in practice, w_{ik} of Eq. (45) is often more efficient for our problems. This interpolation has the set of interpolatory points created only from direct neighbors and this interpolation will have full rank.

5.3. Algorithm for coarsening and defining interpolation weights

For the point $j \in D_i^{\mathrm{S}}$, we need criteria to decide whether the point j has a sufficient negative connection to C_i so that it is distributed to C_i for calculating the interpolation. The accuracy of estimating error e_j ($j \in D_i^{\mathrm{S}}$) by (43) is improved as the negative connection to (or dependence on) the set C_i increases for point j. Therefore it is desirable to have a larger value of $\sum_{l\in C_i} |b_{jl}^-|$. We denote the set of points strongly dependent on the set C_i by S_i^{D}

$$S_i^{\mathrm{D}} = \left\{ j \in D_i^{\mathrm{S}} : \sum_{l\in C_i} |b_{jl}^-| > \varepsilon_{\mathrm{sdp}}(|b_{ij}|/\max|b_{ik}^-|)\max|b_{jl}^-| \quad \text{with fixed } \varepsilon_{\mathrm{sdp}} > 0 \right\}, \tag{46}$$

where $\varepsilon_{\mathrm{sdp}}$ is a constant which defines strong dependence on the set C_i. A large value of $\varepsilon_{\mathrm{sdp}}$ implies that for e_j ($j \in D_i^{\mathrm{S}}$) to be estimated accurately, the points in D_i^{S} need to have a strong dependence on C_i to be included in S_i^{D}. Otherwise, the points in D_i^{S} or point i are put into C to avoid poor estimation of e_j ($j \in D_i^{\mathrm{S}}$). A large value of $\varepsilon_{\mathrm{sdp}}$ increases $|C|$ and this usually increases the computation work. In practice, it is often sufficient if the point j has at least one strong negative connection. $\varepsilon_{\mathrm{sdp}}$ is set to 0.35 for level $l \leqslant 3$ and 0.45 for $l > 3$ in AMGPCG. With these values of $\varepsilon_{\mathrm{sdp}}$, point j usually has at least one strong negative connection for $l \leqslant 3$ and often at least a few strong negative connections for $l > 3$ in C_i for AMGPCG. $\varepsilon_{\mathrm{sdp}}$ is set slightly higher for $l > 3$ because it allows an increase in accuracy of the interpolation for $l > 3$. $|C|$ does not increase greatly because $|C|$ for those levels is already small compare to levels in $l \leqslant 3$.

Because one strong negative connection for the point j to C_i is often sufficient to estimate e_j and it is desirable to distribute C- and F-points reasonably uniformly, Ruge and Stüben [1] suggest the following criteria for choosing the set C and F:

(C1) For each $i \in F$, each point $j \in S_i$ should either be in C, or should be strongly negatively connected to at least one point in C_i ($= C \cap S_i$),
(C2) C should be a maximal subset of all points with the property that no two C-points are strongly negatively connected to each other.

To satisfy the above criteria (C1) and (C2) reasonably, we first use "*Preliminary **C**-point choice*" algorithm described by Ruge and Stüben [1] then the following algorithm is used to select final ***C***-points and define interpolation weights.

*Final **C**-point choice and definition of interpolation weights:*

Step 1. Set $T = \emptyset$.
Step 2. If $T \supseteq F$, stop. Otherwise, pick $i \in F - T$ and $T = T \cup \{i\}$.
Step 3. Set C_i, D_i^{S}, $D_i^{\mathrm{W}-}$, $D_i^{\mathrm{W}+}$, S_i^{D} (see Sections 5.2 and 5.3) and set $\overline{C}_i = \emptyset$.
Step 4. Set $d_i = b_{ii}$ and for $k \in C_i$, $d_k = b_{ik}$.
Step 5. For each $j \in D_i^{\mathrm{W}-}$, $d_i = d_i + b_{ij}$.
Step 6. For each $j \in D_i^{\mathrm{W}+}$, set C_i^{j+} (see Section 5.2)
  if $C_i^{j+} = \emptyset$ then
  $d_i = d_i + b_{ij}$
  else
  $d_k = d_k + b_{ij}b_{jk}^{\mathrm{S}-} / \sum_{l\in C_i} b_{jl}^{\mathrm{S}-}$ for $k \in C_i$.
Step 7. For each $j \in D_i^{\mathrm{S}}$,
  if $j \in S_i^{\mathrm{D}}$ then
  $d_k = d_k + b_{ij}b_{jk}^- / \sum_{l\in C_i} b_{jl}^-$ for $k \in C_i$
  else

if $\overline{C}_i \neq \emptyset$ then
set $C = C \cup \{i\}$, $F = F - \{i\}$ and go to Step 2.
else
set $\overline{C}_i = \{j\}$, $C_i = C_i \cup \{j\}$, $D_i^{\mathrm{S}} = D_i^{\mathrm{S}} - \{j\}$,
update S_i^{D} and go to Step 4.

Step 8. Set $C = C \cup \overline{C}_i$, $F = F - \overline{C}_i$ and $w_{ik} = -d_k/d_i$ for each $k \in C_i$ and go to Step 2.

The effects of assigning the large value for $\varepsilon_{\mathrm{str}}$ at the finest, first and second coarse levels are described in next section.

5.4. Effects of changing ε_{str}

The value of $\varepsilon_{\mathrm{str}}$ is adjusted so that the setup time, AMG cycle time and memory requirement can be reduced without greatly sacrificing convergence behavior. These purposes may be achieved if we can create simpler coarse systems while still using a reasonably accurate interpolation operator.

Whether using a large value of $\varepsilon_{\mathrm{str}}$ in the algorithm for "*Preliminary **C**-point choice*" and "*Final **C**-point choice and definition of interpolation weights*" will increase or decrease $|C|$ is not clear. However our experience shows a tendency for $|C|$ to be decreased for the matrices arising from unstructured 3D meshes and the matrices of the corresponding first few coarser systems. In addition, increasing $\varepsilon_{\mathrm{str}}$ generally implies a smaller number of elements in the set of interpolatory points C_i. Since $|C_i|$ is small and C_i is created only from direct neighbors, the determination of the interpolation weight becomes very simple and therefore, the algorithm of "*Final **C**-point choice and definition of interpolation weights*" can be carried out quickly. Furthermore, a smaller $|C_i|$ implies that the number of nonzero entries of coarser level matrix is smaller. This is because the coarser matrix is constructed using Eq. (8) with the interpolation operator $\mathbf{I}_{l+1}^{l}$ which has a smaller $|C_i|$. Another important benefit of using a large $\varepsilon_{\mathrm{str}}$ is that there is a tendency to have fewer positive off-diagonal entries or smaller values of positive off-diagonal entries in the coarser matrix. This allows the interpolation operator $\mathbf{I}_{l+2}^{l+1}$ to be defined more accurately using our algorithm. Entries of coarser matrix can be expressed as

$$b_{km,l+1} = \sum_{i,j} \bar{w}_{ik} b_{ij,l} \bar{w}_{jm} = b_{km,l} + \sum_{i \in F_l} w_{ik} \left(b_{im,l} + \frac{1}{2} \sum_{j \in F_l} w_{jm} b_{ij,l} \right) + \sum_{i \in F_l} w_{im} \left(b_{ik,l} + \frac{1}{2} \sum_{j \in F_l} w_{jk} b_{ij,l} \right), \quad (47)$$

where $k, m \in C_l$, $\bar{w}_{ik} = \delta_{ik}$ if $i \in C_l$ and $\bar{w}_{ik} = w_{ik}$ if $i \in F_l$. Here δ_{ik} denotes the Kronecker symbol. Usually w_{ik} and w_{im} are positive and a large $\varepsilon_{\mathrm{str}}$ implies generally larger $-b_{im,l}$ and $-b_{ik,l}$. Therefore, $b_{km,l+1}$ tend to become negative or smaller positive for off-diagonal entries. Consequently, applying a very large value of $\varepsilon_{\mathrm{str}}$ for the finest and first few coarse levels can produce substantially simpler and suitable coarse systems and hence reduce the setup time and AMG cycling time while not significantly increasing the convergence factor.

The setup time and AMG cycling time are almost directly related to the total combined number of nonzero entries of matrices of all levels. The memory requirement is closely related to the total number of nonzero entries, although there is also some additional memory required. Unfortunately, a smaller $|C_i|$ often leads to an increase of the convergence factor. However, when C_i is created with a very large value of $\varepsilon_{\mathrm{str}}$, the AMG algorithm coarsens in the direction of very slowly changing algebraically smooth error and tends to produce a coarser matrix with fewer positive off-diagonal entries or smaller values of positive off-diagonal entries. This means interpolation can be reasonably accurate even if we have a small $|C_i|$. The increase of the convergence factor is not substantial so that it is easily overcome by the reduction of necessary computational work and hence the over all benefit is greater. The use of this interpolation is sufficiently accurate when the AMG method is used as a preconditioner for CG. We will see this aspect in the next section.

6. Comparison of test results

Tests for solving various systems of linear equations for pressure were carried out to compare the efficiency of SSORPCG, AMG1R5/SA, AMG1R5/CG, SAMG, AMGPCG0, and AMGPCG1 for 3D problems. In these tests, all AMG solvers are set to use one Gauss–Seidel iteration as a smoother and the V-cycle for the multigrid cycle. Other parameters are optimized to reach convergence within the shortest possible time. The smoother of AMG1R5/CG was modified to perform smoothing with the order of points reversed. This is because the post-smoothing operator $\mathbf{S}_l^{(\mathrm{post})}$ should be adjoint to the pre-smoothing operator $\mathbf{S}_l^{(\mathrm{pre})}$ in the $\mathbf{A}_l$ energy inner product when AMG is used as a preconditioner for CG. Three different finite element models were simulated during the filling phase of polymer injection molding flow simulation and solved at approximately 90% fill by volume. Also for one of the models, we solved the systems of linear equations from the very early stage of filling simulation to demonstrate the efficiency of solving small to medium size problems with our solvers. The convergence tolerances were set to 1.0×10^{-5}, which is the default value used in industry, so that each solver stopped iterating and the elapsed CPU time was measured when the relative residual was reduced to less than this tolerance. The relative residual is defined as the Euclidean norm of the residual vector divided by Euclidean norm of the right hand side vector of Eq. (1) (i.e. $\|\mathbf{f} - \mathbf{A}\mathbf{v}\|/\|\mathbf{f}\|$, where $\mathbf{v}$ is the estimated solution vector).

Two of the test models are plates and their only difference is in the mesh used. The Plate-A problem mostly has elements with relatively small aspect ratios (longest side/shortest side) and the Plate-B problem has many elements with a high aspect ratios (see Figs. 1–3). The main source of ill-conditioning for our

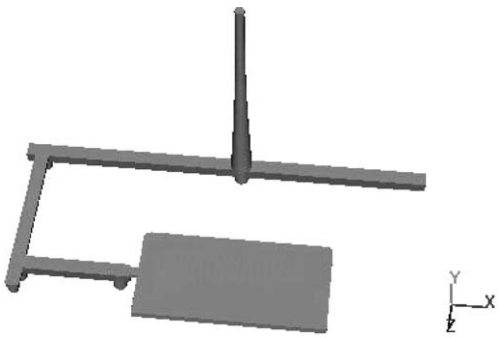

Fig. 1. Plate model.

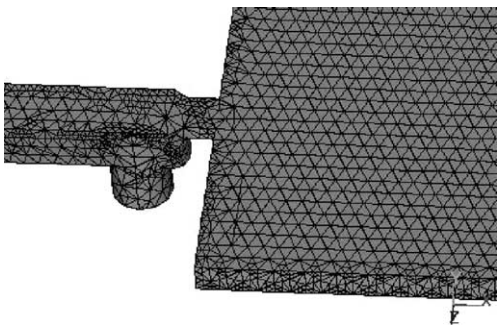

Fig. 2. Mesh Plate-A (small aspect ratio).

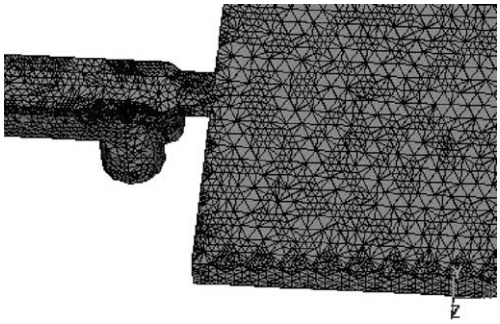

Fig. 3. Mesh Plate-B (large aspect ratios).

problem is due to elements with high aspect ratio. Therefore the Plate-B problem is a good test for the robustness of solvers. The number of nodes of Plate-A and Plate-B are 29,587 and 170,028 respectively. A third test model is an electrical connector problem which has complex geometry (see Fig. 4). This has 200,060 nodes.

Table 1 gives the results for complexities $C_{\mathbf{K}}$ and C_{G}, the elapsed time and the number of iterations. $C_{\mathbf{K}}$ and C_{G} are operator and grid complexities, respectively, defined as

$$C_{\mathbf{K}} = \sum_{l=1}^{l_{\max}} |\mathbf{K}_l|/|\mathbf{A}_1|, \qquad C_{\mathrm{G}} = \sum_{l=1}^{l_{\max}} N_l/N_1, \tag{48}$$

where $|\mathbf{K}_l|$ is the number of nonzero entries for matrix $\mathbf{K}_l$, and N_l is the number of rows of matrix $\mathbf{K}_l$. For AMGPCG0, AMG1R5 and SAMG, $\mathbf{K}_1$ is the original matrix $\mathbf{A}$ (i.e., $\mathbf{A}_1$). For AMGPCG1 $\mathbf{K}_1$ is the approximation of $\mathbf{A}$ (see Eq. (28)). $C_{\mathbf{K}}$ is almost proportional to computational work necessary and approximately proportional to memory required. The setup time and the AMG cycle time are also almost directly related to the value of $C_{\mathbf{K}}$.

In Fig. 5, $\|\mathbf{M}\|$ of AMG scheme of AMGPCG is estimated using error vectors of system of linear equations $\mathbf{Bx} = \mathbf{0}$, where $\mathbf{B}$ is a priori preconditioning matrix. Since the right hand side is zero, the error is equal to the estimated solution. $\|\mathbf{M}\|$ can be expressed as

$$\|\mathbf{M}\| = \sup_{\mathbf{e}^{(m)} \neq \mathbf{0}} (\|\mathbf{M}\mathbf{e}^{(m)}\|/\|\mathbf{e}^{(m)}\|) = \sup_{\mathbf{e}^{(m)} \neq \mathbf{0}} (\|\mathbf{e}^{(m+1)}\|/\|\mathbf{e}^{(m)}\|), \tag{49}$$

where $\|\mathbf{e}^{(m)}\|$ and $\|\mathbf{e}^{(m+1)}\|$ are the Euclidean norm of mth and $m+1$th step errors, respectively.

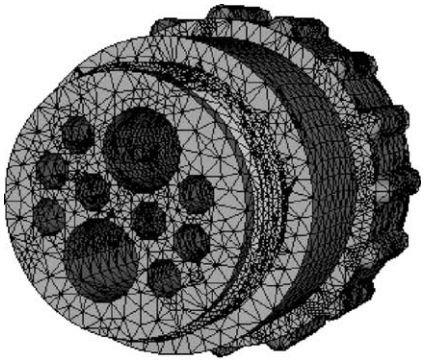

Fig. 4. Electrical connector.

Table 1
Complexities and computation time

Method	Complexities		Elapsed time (s)/Pentium II 233 MHz			Number of iterations
	C_K	C_G	Setup time	Per cycle	Total	
Plate-A						
SSORPCG	1.00	1.00	0.0	0.17	61.0	352
AMG1R5/SA	3.32	2.05	4.4	0.54	12.0	14
AMG1R5/CG	3.32	2.05	4.4	0.66	9.0	7
SAMG	1.23	1.19	1.4	0.36	5.0	10
AMGPCG0	1.74	1.78	1.1	0.30	3.5	8
AMGPCG1	1.41	1.78	1.0	0.26	3.6	10
Plate-B						
SSORPCG	1.00	1.00	0.0	1.1	667.0	604
AMG1R5/SA	4.17	2.24	41.0		Did not converge	
AMG1R5/CG	4.17	2.24	41.0	5.1	143.0	20
SAMG	1.36	1.25	11.0	2.4	42.0	13
AMGPCG0	1.89	1.82	8.0	2.1	29.0	10
AMGPCG1	1.42	1.81	7.0	1.7	33.0	15
Electrical connector						
SSORPCG	1.00	1.00	0.0	1.3	160.0	119
AMG1R5/SA	5.33	2.27	77.0	6.1	426.0	57
AMG1R5/CG	5.33	2.27	77.0	7.3	187.0	15
SAMG	1.38	1.22	15.0	3.0	36.0	7
AMGPCG0	2.03	1.84	11.0	2.8	28.0	6
AMGPCG1	1.61	1.84	10.0	2.3	31.0	9

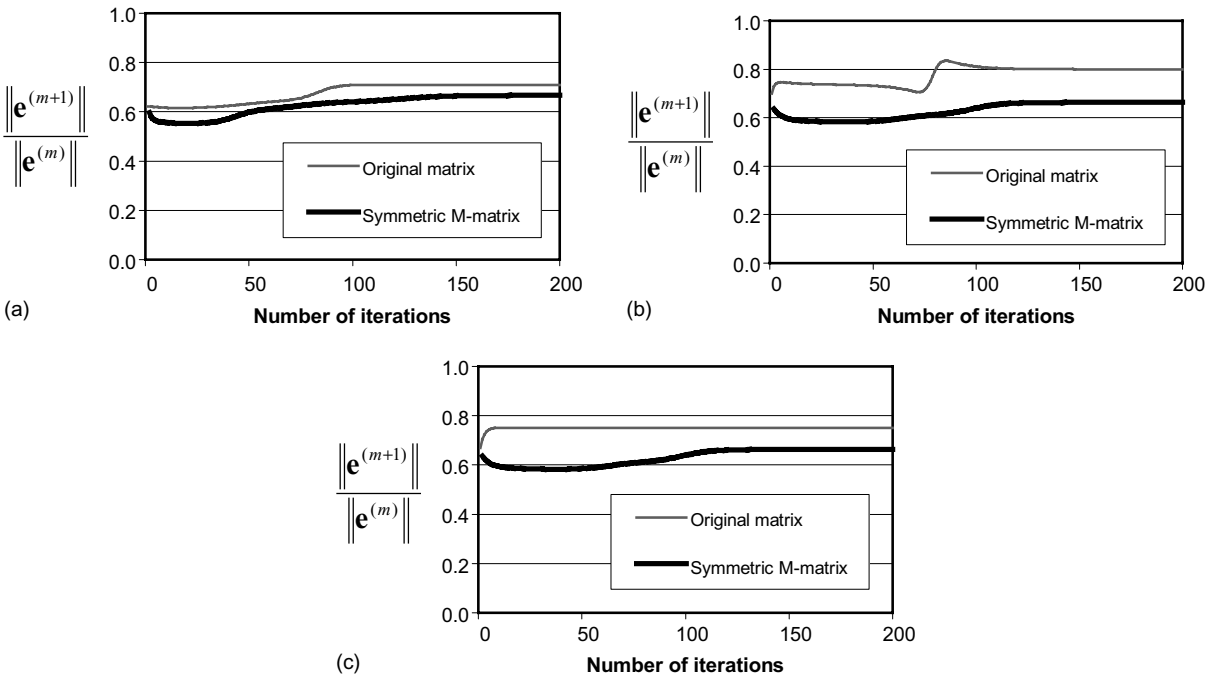

Fig. 5. (a) Estimation of $\|\mathbf{M}\|$ for Plate-A, (b) estimation of $\|\mathbf{M}\|$ for Plate-B and (c) estimation of $\|\mathbf{M}\|$ for the electrical connector.

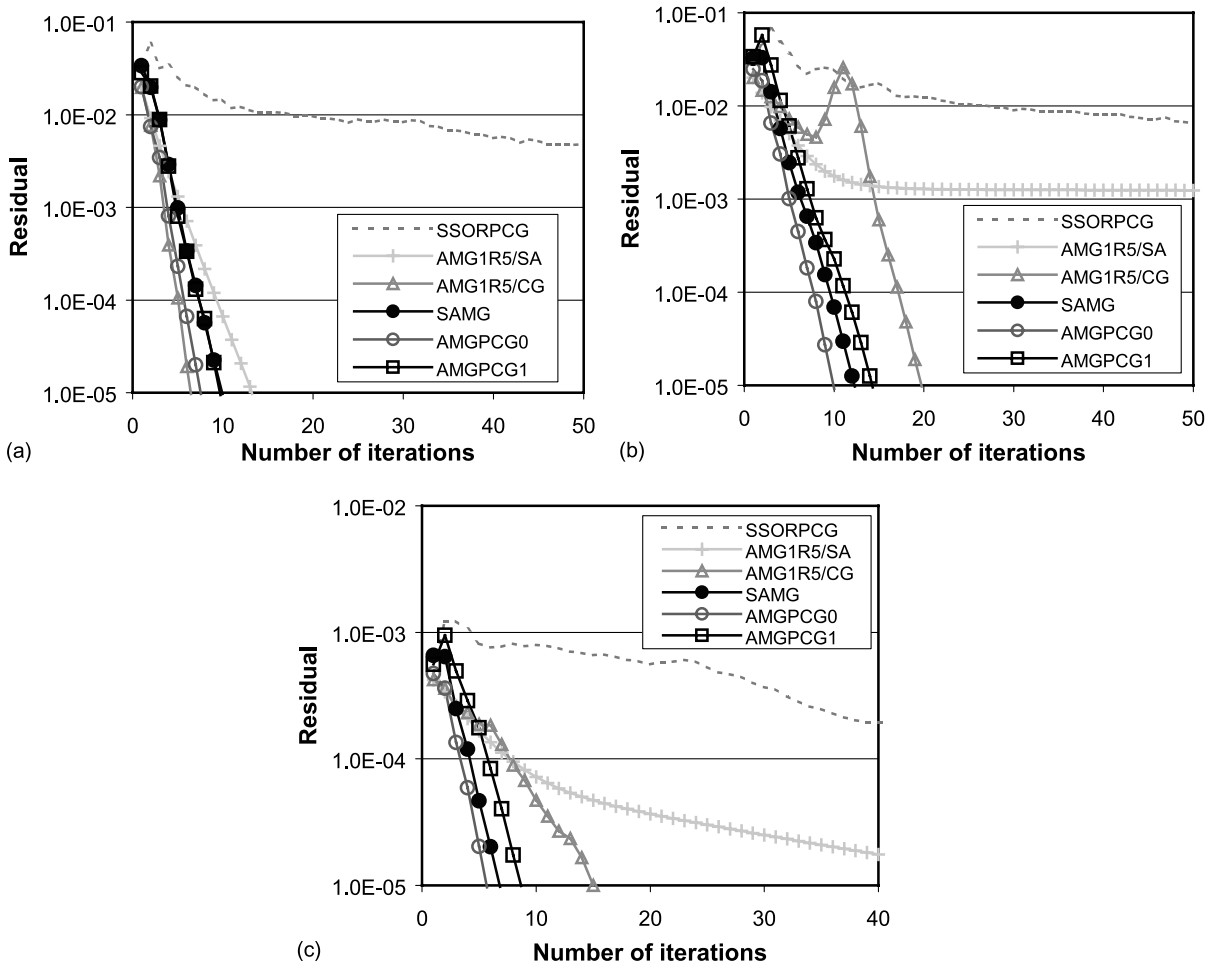

Fig. 6. (a) Histories of relative residual for Plate-A, (b) histories of relative residual for Plate-B and (c) histories of relative residual for the electrical connector.

In Fig. 6 the histories of the relative residuals are shown. In Fig. 7, the cumulative elapsed CPU times are shown for the setup phase and the solution phase to show the efficiency of AMGPCG solvers. Fig. 8 shows the improvement of the elapsed CPU time per solution obtained with AMGPCG for the early stages of filling simulation of the Plate-A (i.e., solving small to medium size systems).

7. Discussion

AMGPCG0 satisfies the assumptions (i)–(iv) rigorously and we have shown $\|\mathbf{M}\|_{\mathbf{B}} < 1$ theoretically in Section 3. Furthermore, we have observed $\|\mathbf{e}^{(m+1)}\|/\|\mathbf{e}^{(m)}\| < 1$ using our AMG scheme for problems arising in polymer injection molding (see also Fig. 5). AMGPCG1 also satisfies the assumptions (i)–(iv) rigorously. We have shown the existence of two positive constants which satisfy the spectral equivalence inequality relation for matrix **B** to **A**. The estimated $\|\mathbf{M}\|$ of the AMG scheme are much smaller when the original matrix is replaced by a symmetric M-matrix (see Fig. 5). This is because the interpolation operator is

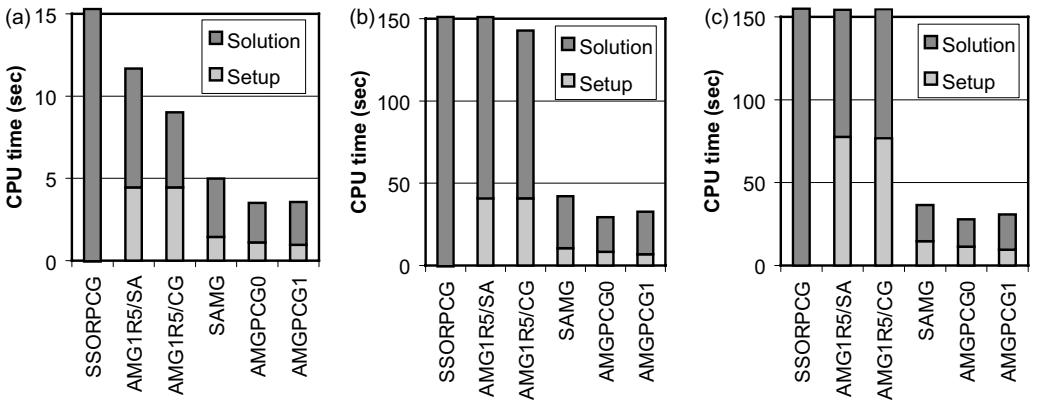

Fig. 7. Elapsed CPU time for (a) Plate-A, (b) Plate-B and (c) electrical connector.

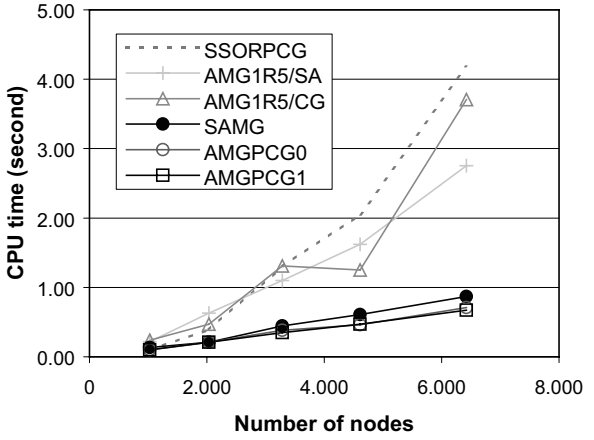

Fig. 8. Elapsed CPU time for small to medium systems for the Plate-A.

defined more accurately and γ of inequality (19) tends to become smaller using our approach. γ tends to become smaller because the diagonal entry becomes larger when there are positive off-diagonal entries in the same row. Therefore, our AMG method is legitimate to use as a preconditioner for CG.

We have tested three problems with nonregular 3D tetrahedral meshes. The Plate-A mostly has elements with small aspect ratios, however some elements are still quite distorted with a large aspect ratio (see Fig. 2). The Plate-B is the most ill-conditioned of the problems tested here with many elements having a large aspect ratio (see Fig. 3). The electrical connector has a complicated geometry and some elements have high aspect ratios (see Fig. 4).

Both AMG1R5/SA and AMG1R5/CG are faster than SSORPCG for the Plate-A problem. However for the Plate-B problem the efficiency of the AMG1R5/SA solver deteriorates significantly and cannot reach the convergence tolerance within 200 iterations. As can be seen from Fig. 6(b), after approximately 15 iterations the residual of AMG1R5/SA does not decrease any further. Even though AMG1R5/CG can reach convergence reasonably fast, we see a fluctuation of the residual from AMG1R5/CG for this problem. This is because the Plate-B problem is ill-conditioned with many positive off-diagonal entries and hence the interpolation weight created by AMG1R5 may not be appropriate. In the case of the electrical

connector problem, AMG1R5/SA can reach the convergence tolerance with 57 iterations, however the residual of AMG1R5/SA has a very gradual decrease. This model has some elements with high aspect ratios (see Fig. 4). Therefore it is also not a well-conditioned problem. AMG1R5/CG reaches convergence in 15 iterations, however, the total time required is greater than SSORPCG. This is particularly due to costly setup time for the AMG algorithm of AMG1R5.

When SAMG is optimized for speed, it reaches convergence very quickly by using "aggressive coarsening" [2] strategy for these problems. This coarsening strategy produces very small operator complexities $C_{\mathbf{K}}$ and grid complexities C_{G} (see Table 1). The operator complexities $C_{\mathbf{K}}$ of SAMG for these problems are 4–15% smaller than $C_{\mathbf{K}}$ of AMGPCG1. The grid complexities C_{G} of SAMG are more than 30% smaller than AMGPCG for these problems. SAMG is significantly faster than SSORPCG and AMG1R5 for these problems. SAMG is much faster than AMG1R5, because the operator complexities are significantly smaller than AMG1R5 while the required number of iterations is low. Smaller operator complexity results in less work for the setup phase and multigrid cycle so that both the setup and multigrid cycle become faster.

Operator complexities of both AMGPCG solvers are also much smaller than the operator complexities of AMG1R5 solvers. Grid complexities of AMGPCG solvers are only slightly smaller than grid complexities of AMG1R5 solvers. This means that a smaller operator complexity is obtained by creating a coarser matrix with fewer nonzero off-diagonal entries rather than reducing $|C|$ substantially. AMG methods with aggressive coarsening, such as SAMG, can produce much smaller operator complexity than our method by reducing $|C|$ substantially (i.e., small C_{G}). Even though our operator complexity is larger than the complexity of SAMG, there are important advantages to our approach. Our approach produces much smaller sets of interpolatory points which are created only from the direct neighbors at the finest and first few coarse levels. Our solvers also have relatively large grid complexity compared to SAMG. Firstly, this means that defining the weight of interpolation is simple using our algorithm so that setup is much faster in spite of larger complexity. Secondly, during the solution phase, a larger operator complexity means that the work necessary for smoothing is larger. However, the much smaller sets of interpolatory points at those levels and relatively large grid complexity reduces the required computational work for interpolation (bringing the solution of a coarse system to a fine system) and restriction (bringing the residual of a fine system to a coarse system). Hence using our approach, the over all computational work for the multigrid cycle can be less in spite of larger operator complexity. Furthermore, we use single precision variables to estimate the solution of the a priori preconditioning system (4). This reduces the memory requirement for our AMG scheme. It also reduces setup time and AMG cycle time approximately 10–20% compared to using double precision variables when it is tested on our 32-bit hardware. Therefore the setup and multigrid cycle of AMGPCG are fast (see Table 1).

Both the AMGPCG solvers have a rapid setup and can reach a convergence tolerance within a small number of iterations (mostly less than 20 iterations) and have a steady residual decrease for any problems we have tested. AMGPCG1 slightly increases the number of necessary iterations compared to AMGPCG0. However as seen from Table 1, setup time and cycle time are fast so that the total time required is approximately the same or only slightly increases compared to AMGPCG0 for these problems. This is because AMGPCG1 has a smaller operator complexity. It is approximately 75–82% of AMGPCG0 for these problems. In some cases, particularly when the model mostly has elements with small aspect ratios, AMGPCG1 converged faster than AMGPCG0. This is because the positive off-diagonal entries of the original system matrix tend to become smaller and hence the ratio of two constants $\alpha_{\mathbf{B}}$ and $\beta_{\mathbf{B}}$ of inequality (12) $\beta_{\mathbf{B}}/\alpha_{\mathbf{B}}$ becomes smaller. This means AMGPCG1 hardly increases the number of necessary iterations compared to AMGPCG0 while its operator complexity is smaller than the operator complexity of AMGPCG0.

The AMGPCG solvers are always the fastest among the solvers we have tested here. For the ill-conditioned problem of Plate-B, both the AMGPCG solvers are more than 20 times faster than SSORPCG, more than four times faster than AMG1R5/CG and slightly faster than SAMG. Our AMGPCG is much

faster than AMG1R5/CG because the setup and each cycle times are very fast and the interpolation weight is set appropriately (i.e., there are no large fluctuations of residual as observed in AMG1R5/CG). For the Plate-B problem, many interpolation weights become negative for AMG1R5 while no negative interpolation weights occurred for AMGPCG. Also AMGPCG is faster than SAMG for these problems because setup and cycle time are faster while the required number of iterations are similar to SAMG. Even though SAMG is slightly slower than AMGPCG for these problems, its memory requirement is smaller than AMGPCG. Furthermore SAMG is very general. It can efficiently solve many different types of matrices. All solvers use the previous time step's solution as an initial estimate for the current step. The residual is only required to reduce by two orders of magnitude for the electrical connector problem, and approximately four orders of magnitude for the Plate-A and Plate-B problems in order to satisfy the convergence tolerance. Therefore fast setup is very important for these problems. Both AMGPCG solvers complete the calculation to estimate the solution even before the setup phase of AMG1R5 has been completed for these three problems (see Table 1 and Fig. 7).

Fig. 8 shows the total time required to solve a system of linear equations for pressure at the early stages of polymer injection flow simulation. As can be seen from Fig. 8, both SAMG and the AMGPCG solvers are very fast compared to other solvers. When the system has 2000 nodes, both SAMG and AMGPCG are already faster than SSORPCG and at 6000 nodes, these solvers are approximately five times faster than SSORPCG and three to four times faster than both AMG1R5 solvers. Therefore both SAMG and the AMGPCG solvers can be consistently used for any matrix problem size.

Often in industrial problems, including polymer injection molding flow simulation, a very high-accuracy approximation is not required. Therefore the value of tolerance is not set to a very small value. Furthermore, a reasonably accurate estimate is often available from the previous time step. Therefore an AMG solver with fast setup phase is ideally suited to this kind of simulation.

8. Conclusions

Systems of linear equations arising from the finite element discretization of the pressure equation on unstructured meshes in polymer injection molding have been solved by SSORPCG, AMG1R5/SA, AMG1R5/CG, SAMG, AMGPCG0, and AMGPCG1 to demonstrate that the our approach of using AMG as a preconditioner for CG can produce a very efficient and robust solver.

Firstly, robustness is improved substantially by creating an appropriate interpolation operator for the AMG method and using this AMG method as a preconditioner for CG.

Secondly, simpler coarse systems (smaller operator complexity) are obtained by reducing the first few coarse systems substantially by using a very small $|C_i|$ where C_i is only created from the very strongly negatively connected direct neighbors. In the case of AMGPCG1, further reduction in operator complexity is obtained by a simple approximation of the original system matrix for the a priori preconditioning matrix. In addition, our coarsening strategy is simple. Hence we can accomplish the setup phase very quickly and the use of our simple coarse systems provides fast iteration cycles while not greatly increasing the convergence factor.

Consequently, our solvers are very robust and efficient for solving any size of systems of linear equations arising from discretization of a scalar elliptic PDE on unstructured 3D meshes. Also, our solvers can efficiently solve problems requiring only low-accuracy approximations.

Acknowledgements

This project is carried out under the CRC-IMST program with the Industrial Research Institute Swinburne and Moldflow Pty Ltd. The authors would like to express their appreciation to Moldflow Pty Ltd.

and the Industrial Research Institute Swinburne for providing funding and equipment, and to the Moldflow 3D Research Group and Zhiliang Fan of the Moldflow Solver Research Group in particular for advice in many significant areas.

References

[1] J. Ruge, K. Stüben, Algebraic multigrid, in: S. McCormick (Ed.), Multigrid methods, Frontiers in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1987.
[2] K. Stüben, Algebraic Multigrid (AMG): An Introduction with Applications, GMD Report 70, Sankt Augastin, Germany, 1999.
[3] L. Hageman, D. Young, Applied Iterative Methods, Academic Press, New York, 1981.
[4] W. Briggs, V. Henson, S. McCormick, A Multigrid Tutorial, second ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 2000.
[5] W. Hackbusch, Multi-grid Methods and Applications, Springer-Verlag, Berlin, 1985.
[6] M. Jung, U. Langer, A. Meyer, W. Queck, M. Schneider, Multigrid preconditioners and their applications, in: G. Telschow (Ed.), Third Multigrid Seminar, Akademie der Wissenschaften der Ddr, Berlin, 1989.
[7] O. Axelsson, Iterative Solution Methods, Cambridge University Press, Cambridge, 1994.