INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **53**:669–690
Published online 6 July 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1297

Algebraic multigrid and incompressible fluid flow

R. Webster*, †

Roadside, Harpsdale, Halkirk, Caithness, Scotland, U.K.

SUMMARY

This paper is concerned with the development of algebraic multigrid (AMG) solution methods for the coupled vector–scalar fields of incompressible fluid flow. It addresses in particular the problems of unstable smoothing and of maintaining good vector–scalar coupling in the AMG coarse-grid approximations. Two different approaches have been adopted. The first is a direct approach based on a second-order discrete-difference formulation in primitive variables. Here smoothing is stabilized using a minimum residual control harness and velocity–pressure coupling is maintained by employing a special interpolation during the construction of the inter-grid transfer operators. The second is an indirect approach that avoids the coupling problem altogether by using a fourth-order discrete-difference formulation in a single scalar-field variable, primitive variables being recovered in post-processing steps. In both approaches the discrete-difference equations are for the steady-state limit (infinite time step) with a fully implicit treatment of advection based on central differencing using uniform and non-uniform unstructured meshes. They are solved by Picard iteration, the AMG solvers being used repeatedly for each linear approximation.

Both classical AMG (C-AMG) and smoothed-aggregation AMG (SA-AMG) are used. In the direct approach, the SA-AMG solver (with inter-grid transfer operators based on mixed-order interpolation) provides an almost mesh-independent convergence. In the indirect approach for uniform meshes, the C-AMG solver (based on a Jacobi-relaxed interpolation) provides solutions with an optimum scaling of the convergence rates. For non-uniform meshes this convergence becomes mesh dependent but the overall solution cost increases relatively slowly with increasing mesh bandwidth. Copyright © 2006 John Wiley & Sons, Ltd.

Received 8 February 2006; Revised 11 May 2006; Accepted 13 May 2006

KEY WORDS: algebraic multigrid; fully coupled solutions; Navier–Stokes; mimetic methods

1. INTRODUCTION

Since the introduction of algebraic multigrid (AMG) over twenty years ago [1–4], it has become an established method for the numerical solution of scalar, elliptic, partial differential equations,

*Correspondence to: R. Webster, Roadside, Harpsdale, Halkirk, Caithness, Scotland, U.K.
†E-mail: ronniewebster@aol.com

especially for geometrically complex domains discretized using unstructured grids. For these and other applications, it has provided a robust efficient mesh-independent convergence (see the review by Stueben [5, 6]). Despite this, there have been applications where the method has not provided the expected performance and this has prompted a resurgence of research into AMG [7, 8]. One area which features in this research programme is the application of the method to systems of partial differential equations, such as the coupled, vector–scalar, system of fluid velocity and pressure as described by the Navier–Stokes equations. The second-order discrete-difference formulation for this system (referred to here as PV2) is one of the systems to be investigated. The application of C-AMG solvers to advection–diffusion problems [6] and to the fully coupled PV2 systems (see below) can provide good convergence characteristics, however, the scaling of the convergence can exhibit a mesh dependence. A coupled SA-AMG solver for PV2 systems which exhibits almost mesh-independent scaling has been reported [9, 10], but not the treatment of velocity–pressure coupling used. There seems to be no published work on the application of C-AMG-type solvers to such systems. An alternative formulation of the equations for incompressible fluid flow is the fourth-order, discrete stream-function formulation of Chang *et al*. [11]. At the time of writing, there seem to be no reported applications of AMG to this system (here referred to as SF4).

SA-AMG and C-AMG solutions for both PV2 and SF4 systems are investigated in this work with a view to addressing the problem of mesh-dependent convergence, thought to be due to a poor representation of the inter-field coupling in the coarse-grid approximations (CGAs). Before proceeding, the background to the discrete stream-function (incompressible basis function) method SF4, is outlined.

Staggered-grid methods on Cartesian meshes, as developed by Harlow and Welch [12], featured prominently in the early development of numerical solution methods for incompressible fluid flow, and are still in widespread use. The popularity of the approach is due largely to its numerical stability and to the early recognition that it is globally conservative [13]. In recent years there has been a revival of interest in staggered-grid discretization schemes, and in particular for extending the method to general, unstructured, grids. This is partly due to recognized weaknesses in the stability of collocated discretizations for unstructured grids (see, for example, Reference [10]), and partly due to progress made in extending the staggered-grid method to unstructured triangular meshes [14, 15]. Further impetus has been gained from the development of mimetic finite difference methods [16–18], where the objective is a self-consistent, discrete, vector and tensor calculus that 'mimics' the vector and tensor calculus of the continuum [18]. A truly mimetic discretization produces solutions that automatically satisfy the fundamental conservation laws both locally and globally. The discrete-differential operators and their derived adjoint counterparts, the so-called support operators, fall naturally on complementary staggered or dual domains, so that higher-order compound operators may be formed naturally, the range of the first operator matching the domain of the second [17]. A discrete vector calculus for tensor product grids, for example, produces a staggered grid system similar to that of Harlow and Welch. Successful, conservative, mimetic discretizations using the support operator approach have been achieved for general, logically rectangular, grids in the fields of compressible hydrodynamics (see, for example, Reference [19]) and in electromagnetism [20].

Unfortunately, application of the mimetic approach to general unstructured grids can result in implicit, non-local, support operators [17]. Only in special cases are the derived adjoints explicit and local, e.g. orthogonal tensor product grids and Delaunay grids. This does not prevent solutions for unstructured grids, since matrix-free Krylov methods can be employed. However, it would

DOI: 10.1002/fld

preclude the use of those iterative methods requiring an explicit system matrix, such as AMG methods, and hence also the prospect of a mesh-independent convergence.

The unstructured, staggered-grid, scheme of Perot [21] is not formally derived using the support operator method, but it does use dual grids and discrete vector operators that are constructed to satisfy the vector identities of their continuum counterparts. It is also mimetic in the sense that it has been shown to possess the required conservation properties. When applied to the Navier–Stokes equations in primitive-variable form, the scheme has been shown to conserve mass, momentum and energy and, when cast in rotational form, it has also been shown to conserve vorticity [21]. The discrete equivalents of the vector operators Grad, Div, Curl and Rot may also be used to transform and reduce the coupled system to a single-equation system for one, scalar, field variable, a discrete stream-function [11]. The pressure/continuity equation is thereby eliminated, incompressibility now being an intrinsic property of stream-function space. Moreover, the discrete stream-function variable, the vector potential integrated along a cell edge, is a scalar for both 2D and 3D meshes. The scheme does deliver an explicit system matrix, so AMG linear solvers may be used. However, the penalty paid for the reduction is a higher, fourth-order, discrete-difference operator. This will impact on the required inter-grid transfer operators and the CGAs of AMG.

2. DISCRETIZATIONS

Both PV2 [10, 22] and SF4 [11, 21] discretizations are derived using finite-volume methods on unstructured finite-element meshes. As the enforcement of the relevant conservation laws for the chosen control cells is a standard procedure it will not be described in any detail here. In both cases (PV2 and SF4) it results in two matrix equations in three vector unknowns. The focus is on the methods of closure, and hence on the final forms to be solved by AMG.

2.1. Primitive-variable formulation; PV2

Fluid velocity, $\mathbf{v}$, and the pressure, p, are both defined at element vertices. The control cell is thus the median dual enclosing each vertex. An additional fluid velocity within elements, $\mathbf{v}^e$, is defined at element centres. Enforcing the conservation laws consistently for all control cells, using $\mathbf{v}^e$ to evaluate the fluxes for the control surfaces within elements, delivers a coupled system of equations:

$$\begin{aligned} \mathbf{Q}(\mathbf{v}^e)\mathbf{v} + \mathbf{G}^n\mathbf{p} &= \mathbf{b} \\ -\mathbf{D}\mathbf{v}^e &= \mathbf{0} \end{aligned} \tag{1}$$

where $\mathbf{Q}$ is the advection–diffusion operator, $\mathbf{D}$ is the element-to-node divergence operator and $\mathbf{b}$ is the momentum source for the cells. Note that $\mathbf{v}$ and $\mathbf{v}^e$ are vectors of flow vectors and $\mathbf{p}$ is a vector of scalar pressures. $\mathbf{G}^n$ is a node-to-node gradient operator derived from $-\mathbf{D}^T$ assuming the pressure interpolates linearly within elements. The entries in $\mathbf{Q}$ are fluxes: those in $\mathbf{D}$ and $\mathbf{G}^n$ are vector areas. To close the system, the velocity, $\mathbf{v}^e$, needs to be expressed in terms of the nodal fluid velocities and pressures, $\mathbf{v}$ and $\mathbf{p}$. To facilitate this a second control cell is constructed within each element and again the conservation laws consistently enforced (see, for example, Reference [10]). This gives the additional equation set

$$\mathbf{Q}^e(\mathbf{v}^e)\mathbf{v}^e - \mathbf{F}(\mathbf{v}^e)\mathbf{v} - \mathbf{G}\mathbf{p} = \mathbf{b}^e \tag{2}$$

DOI: 10.1002/fld

where $\mathbf{Q}^{\mathrm{e}}(\mathbf{v}^{\mathrm{e}})$ is the element-to-element advection–diffusion operator, $\mathbf{F}(\mathbf{v}^{\mathrm{e}})$ is the node-to-element advection–diffusion operator and $\mathbf{G}$ is the node-to-element gradient operator ($\mathbf{G} = -\mathbf{D}^{\mathrm{T}}$); $\mathbf{b}^{\mathrm{e}}$ is the vector of momentum sources for elements. Since there is no direct coupling between elements (only indirect coupling via nodes), $\mathbf{Q}^{\mathrm{e}}(\mathbf{v}^{\mathrm{e}})$ is diagonal. Taking linear approximations, by setting $\mathbf{v}^{\mathrm{e}}$ in both $\mathbf{Q}^{\mathrm{e}}$ and $\mathbf{F}$ to the current value, the solution for an updated $\mathbf{v}^{\mathrm{e}}$ is straight forward:

$$\mathbf{v}^{\mathrm{e}} = (\mathbf{Q}^{\mathrm{e}})^{-1}\{\mathbf{b}^{\mathrm{e}} + \mathbf{F}\mathbf{v} + \mathbf{G}\mathbf{p}\} \tag{3}$$

Substitution in (1) gives the coupled set

$$\begin{aligned} \mathbf{Q}(\mathbf{v}^{\mathrm{e}})\mathbf{v} + \mathbf{G}^{\mathrm{n}}\mathbf{p} &= \mathbf{b} \\ \mathbf{D}^{\mathrm{n}}\mathbf{v} + \mathbf{B}\mathbf{p} &= \mathbf{S} \end{aligned} \tag{4}$$

with $\mathbf{S} = -\mathbf{D}(\mathbf{Q}^{\mathrm{e}})^{-1}\mathbf{b}^{\mathrm{e}}$, $\mathbf{D}^{\mathrm{n}} = \mathbf{D}(\mathbf{Q}^{\mathrm{e}})^{-1}\mathbf{F}$, a node-to-node divergence operator and $\mathbf{B} = \mathbf{D}(\mathbf{Q}^{\mathrm{e}})^{-1}\mathbf{G} = -\mathbf{G}^{\mathrm{T}}(\mathbf{Q}^{\mathrm{e}})^{-1}\mathbf{G}$, a node-to-node Poisson-type operator with nearest-neighbour coupling. The system should therefore be stable and should not normally be susceptible to checkerboard-type pressure instabilities (however, see Reference [10]). It is solved by Picard iteration, in which repeatedly updated, and again linearized, approximations are solved by AMG. The linearization is obtained by setting $\mathbf{v}^{\mathrm{e}}$ to its current value from (3).

2.2. Stream-function formulation, SF4

In this case the control cell is the element itself. Face-normal flux, $\mathbf{U}$, is defined at the face centres. Pressure, p, is defined at the cell centre and a cell fluid velocity, $\mathbf{v}^{\mathrm{c}}$, is also defined at the cell centre. Thus, enforcing the conservation laws for the cells delivers the system

$$\begin{aligned} \mathbf{A}(\mathbf{v}^{\mathrm{c}})\mathbf{v}^{\mathrm{c}} + \mathbf{G}^{\mathrm{c}}\mathbf{p} &= \mathbf{b} \\ -\mathbf{D}\mathbf{U} &= \mathbf{0} \end{aligned} \tag{5}$$

where $\mathbf{A}$ is the advection–diffusion operator for cells, $\mathbf{G}^{\mathrm{c}}$ is the cell-to-cell gradient operator, $\mathbf{D}$ is the face-to-cell divergence operator with non-zero entries of ± 1, $\mathbf{U}$ is the vector of scalar face-normal fluxes, $\mathbf{p}$ is the vector of scalar cell pressures and $\mathbf{b}$ is the vector of the cell sources of momentum. To close this system it is necessary to express the cell fluid velocity, $\mathbf{v}^{\mathrm{c}}$, in terms of the face-normal fluxes, $\mathbf{U}$. This is accomplished using Perot's reconstruction, $\mathbf{R}$, where

$$\mathbf{v}^{\mathrm{c}} = \mathbf{R}\mathbf{U} \tag{6}$$

where for just one cell

$$\mathbf{v}^{\mathrm{c}} = \sum_{i=1}^{i=3} \frac{\mathbf{r}_i U_i}{V_{\mathrm{c}}} \tag{7}$$

where $\mathbf{r}_i$ is the centre-cell to centre-face vector (index i) and V_{c} is the cell volume. Combining (7) with a linearized form of momentum equation in (5) gives

$$\begin{aligned} [\mathbf{A}(\mathbf{v}^{\mathrm{c}})\mathbf{R}]\mathbf{U} + \mathbf{G}^{\mathrm{c}}\mathbf{p} &= \mathbf{b} \\ -\mathbf{D}\mathbf{U} &= \mathbf{0} \end{aligned} \tag{8}$$

DOI: 10.1002/fld

To complete this system the momentum equations are averaged by integration to the faces of cells using the transpose of **R**,

$$\begin{aligned}[\mathbf{R}^{\mathrm{T}}\mathbf{A}\mathbf{R}]\mathbf{U}+\mathbf{G}\mathbf{p}&=\mathbf{R}^{\mathrm{T}}\mathbf{b}\\ -\mathbf{D}\mathbf{U}&=\mathbf{0}\end{aligned} \tag{9}$$

where **G**, a revised gradient operator, satisfies $\mathbf{G}=\mathbf{R}^{\mathrm{T}}\mathbf{G}^{\mathrm{c}}=-\mathbf{D}^{\mathrm{T}}$. This discrete coupled system, which contains the boundary conditions, may be reduced to a single system using one further transformation based on the discrete equivalents of Curl and Rot, **C** and $\mathbf{C}^{\mathrm{T}}$, respectively. The cell edge-to-face Curl operator, **C**, relates the face-normal fluxes, **U**, to integrated stream-functions, **s**, for bounding edges. Thus, for each edge a scalar integrated stream-function, s, is defined as

$$s=\int_{\mathrm{edge}}\boldsymbol{\Psi}\bullet\mathbf{dl} \tag{10}$$

for vector stream-function $\boldsymbol{\Psi}$. The global discrete Curl operator, **C**, relates the vector of scalar face-normal fluxes **U** to the vector of scalar discrete stream-functions, **s**, i.e.

$$\mathbf{U}=\mathbf{C}\mathbf{s} \tag{11}$$

C and $\mathbf{C}^{\mathrm{T}}$, like **G** and **D** contain non-zero entries of ±1. They are truly mimetic in that they satisfy the discrete equivalents of the well-known vector identities, $\mathbf{D}\mathbf{C}=0$ and $\mathbf{C}^{\mathrm{T}}\mathbf{G}=0$, see Reference [11]. Substituting (11) into (9) and applying $\mathbf{C}^{\mathrm{T}}$ gives the transformed system. However, the off-diagonal blocks contain the above identities, and hence the system reduces to a single block

$$[(\mathbf{R}\mathbf{C})^{\mathrm{T}}\mathbf{A}(\mathbf{v}^{\mathrm{c}})\mathbf{R}\mathbf{C}]\mathbf{s}=(\mathbf{R}\mathbf{C})^{\mathrm{T}}\mathbf{b} \tag{12}$$

The transformed matrix shares the numerical properties of **A** [11]. It is the linearized form that is solved by AMG. From the solution, **s**, face fluxes, **U**, are recovered using (11) and cell velocities, $\mathbf{v}^{\mathrm{c}}$, using (6). The linearized form is obtained by using previous iterate values for $\mathbf{v}^{\mathrm{c}}$ in $\mathbf{A}(\mathbf{v}^{\mathrm{c}})$. Note that the entries in **s** are scalars for both 2D and 3D.

3. ALGEBRAIC MULTIGRID SOLVERS

AMG methods are founded on the two basic multigrid principles of error smoothing and coarse-grid correction. Short-range (high wave-number) errors may be smoothed by simple relaxation and long-range (low wave-number) errors can be well represented, and therefore corrected from coarser grids. By careful and complementary enforcement of these two principles on a full hierarchy of successively coarser grids, full bandwidth corrections may be assembled and applied in an iterative process to achieve efficient solutions at a mesh-independent convergence rate. In AMG (in contrast to geometric multigrid, GMG) the simplest possible smoothing is adopted and the coarsening then constructed so as to complement that smoothing using a fully automatic, algebraic, procedure based on information contained entirely in the entries of the system matrix.

Both the PV2 and the SF4 formulations present significant difficulties when trying to satisfy these two basic multigrid principles. In the case of PV2, the numerical properties of the coupled system matrix can compromise the stability of simple relaxation smoothing. Also, the coarsening that is based on the intra-field coupling (off-diagonal entries of the diagonal blocks) may not result

DOI: 10.1002/fld

in the best representation of pressure–velocity coupling (entries in the off-diagonal blocks) of the CGA. The latter problem is avoided in the single-block SF4 formulation, but the system matrix in this case presents its own difficulties. It represents a higher, fourth-order, discrete-difference operator, and therefore demands higher-order inter-grid transfer operators.

Before describing the methods adopted for addressing these difficulties, the two AMG methods used will be outlined. Both are based on the Galerkin CGA, but distinguished by different coarsening schemes, the aggregation-based scheme of Vanek *et al*. [23] and the classic C–F partitioning scheme of Ruge and Stueben [4]. Apart from these differences the solvers are identical. The use of the labels SA-AMG and C-AMG solvers should not be taken to mean that they are representative of all those in the general classifications of smoothed-aggregation and classical AMG. It should also be noted at this point, that in previously reported work [9], it was wrongly implied that the SA-AMG solver was deficient when applied to the cases of low Reynolds number flow on highly stretched grids. It has since been found that this implication is false. Poor performance in those cases was due to a failure in the stability of the discretization itself and not due to any inherent deficiency in SA-AMG for coupled systems [10].

Both solvers were constructed following the guidelines set out by the originators, Vanek *et al*. in the case of SA-AMG [23], Ruge, Stueben, McCormick and others in the case of C-AMG [4, 6, 7]. Since it is the application of the two AMG approaches to the PV2 formulation and the SF4 formulation of the flow equations that is particular interest here, only those aspects are described that are considered especially important in these applications. Other aspects of the methodology and the algorithms are well documented in current research papers [7] and in books [6].

In both applications, the formation of the inter-grid transfer operators, and hence the CGA is particularly important to the efficiency of the solution, and it is in this area that attention is first focussed. Much of the current research on AMG is focussed in this area also [24]. However, attention then turns to the method of pre-smoothing and post-smoothing the transferred residuals and corrections, which is particularly important in the case of the coupled system [9, 22].

3.1. Coarsening for SA-AMG

The SA-AMG solver for coupled fields follows the 'unknown' approach of Ruge and Stueben [4] where for each unknown, regardless of field identity, we associate one algebraic grid point. Thus, for a collocated mesh of N vertices there may be as many as $(d+1) \times N$ fine-grid points for a primitive-variable formulation, where d is the number of spatial dimensions. In SA-AMG the grid points are decomposed into disjoint aggregates such that points within any one aggregate share both the same field and a common, strongly coupled, neighbourhood within that field. A coupling, a_{ij}, between two grid points, i and j, is strong if

$$|a_{i,j}| > \xi_{\mathrm{st}} \sqrt{a_{ii} a_{jj}} \tag{13}$$

where ξ_{st} is a predefined strong coupling parameter. For every fine-grid point there is just one coarse-grid point, and this defines a zero-order interpolation/restriction operator, $\mathbf{I}$, with positive non-zero entries of unity. Because coarse-grid points have the same field identity as their associated fine-grid aggregates, the matrix has block diagonal structure. Note that the coarsening rate for each set of field equations may differ. To improve the CGA, the blocks of $\mathbf{I}$ are smoothed using Jacobi relaxation. This enhances the interpolation, and hence the consistency of the CGA. Depending on the quality of the aggregation, smoothing should produce an approximate first-order interpolation.

DOI: 10.1002/fld

The coarse-grid generation algorithm used is given in Reference [9]; the simpler of the two aggregation algorithms presented there is the one actually used here.

3.2. Coarsening for C-AMG

The C-AMG solver also follows the unknown approach of Ruge and Stueben. Coarse grids are derived from a partitioning of points, Ω^h, on a fine-grid level, h, into two disjoint sub-sets, C^h and F^h, where

$$\Omega^h = C^h \cup F^h$$

and where the coarse-grid points for level, H, are set to be $\Omega^H = C^h$. The partitioning is made using a two-pass scheme, the first pass following Stueben's algorithm [6], the second following that of McCormick [25]. They are controlled by the two heuristics:

1. For every F point, i, every point, j, directly connected to i that strongly influences i should be in set C, or should strongly depend on at least one point in C.
2. The set C should be a maximal sub-set such that no point in C depends strongly on another C point.

The first heuristic is enforced in the interests of a good interpolation; the second is used as a guide in order to achieve a significant dilution of points for the coarse grid, thereby constraining the overall complexity of the CGA. In this case the strong influences and dependencies are defined in terms of the strong coupling parameter, ξ_{st}, as

$$-a_{ij} \geqslant \xi_{\mathrm{st}} \max_{k \neq i}(-a_{ik}) \tag{14}$$

where, again, a_{ij} is the coupling strength between points i and j.

Having established the coarse grid, level H, it is then necessary to define the interpolation, I_H^h, to level h. The approach adopted here follows largely that described by Stueben [6] for essentially positive matrices where any positive off-diagonal entries are considered weak coupling. No advantage was gained by allowing for the separate treatment strong positive coupling. From this point the level superscripts are taken as understood in the interests of simplicity. For the C points on Ω the interpolation is a simple injection. For the F points, i, it is necessary to determine interpolation weights, ω_{ij}, $j \in C$. Those modes of the error spectrum that require correction from the coarse grids, are those which are seen as smooth by Gauss–Seidel/Jacobi relaxation operators on the fine grid, that is, those for which residuals are small. If $\mathbf{e}$ represents the error vector, then for an F point, i, we then have

$$a_{ii}e_i + \sum_{j \in N_i} a_{ij}e_j \approx 0 \tag{15}$$

where the set, N_i, represents neighbours of i. If we define C_i and F_i as

$$C_i = N_i \cap C; \quad F_i = N_i \cap F; \quad N_i = C_i \cup F_i$$

then following Stueben [3], Equation (15) may be approximated by

$$\hat{a}_{ii}e_i + \alpha \sum_{j \in N_i} a_{ij}e_j = 0, \quad \text{with } \hat{a}_{ii} = a_{ii} + \sum_{j \notin P_i} a_{ij}, \quad \alpha = \frac{\sum_{j \in N_i} a_{ij}}{\sum_{j \in P_i} a_{ij}} \tag{16}$$

DOI: 10.1002/fld

where $P_i \subset C_i$ a chosen selection of C neighbours for interpolation. This gives a *direct interpolation*

$$e_i = \sum_{k \in P_i} \omega_{ik} e_k \quad \text{with } \omega_{ik} = -\alpha \frac{a_{ik}}{\hat{a}_{ii}} \tag{17}$$

The choice P_i is usually those members of C_i that are strongly negatively coupled to i. This approximation is insufficient for some of the applications considered here. Better approximations may be obtained by replacing all e_j, $j \in F_i$ in (15) with

$$e_j = \sum_{k \in N_j} \frac{a_{jk} e_k}{a_{jj}} \tag{18}$$

augmenting, thereby, the set of C-points for interpolation. Following Stueben [6], this will be referred to as *standard interpolation*. Allowance for a further extension of the interpolation is also made by including a further layer of contributing points so that

$$e_j = \sum_{k \in C_j} \left(\frac{a_{jk} e_k}{a_{jj}} \right) + \sum_{k \in F_j} \left\{ a_{jk} \sum_{q \in N_k} \left[\frac{a_{kq} e_q}{a_{jj} a_{kk}} \right] \right\} \tag{19}$$

This will be referred to as Jacobi-relaxed interpolation. The extra layer need only be included if the initial F–F coupling is sufficiently strong. Following these substitutions, (18), (19) or both, collecting and summing coefficients with common indices delivers a new equation for e_i

$$\hat{a}_{ii} e_i + \sum_{j \in \hat{N}_i} \hat{a}_{ij} e_j \approx 0, \quad \hat{N} = \{ j \neq i : \hat{a}_{ij} \neq 0 \} \tag{20}$$

which may be used, just as was (15), to obtain a new interpolation.

$$e_i = \sum_{k \in P_i} \omega_{ik} e_k \quad \text{with } \omega_{ik} = -\alpha \frac{\hat{a}_{ik}}{\hat{a}_{ii}} \tag{21}$$

The choice for P_i has been found to be particularly important for solutions of the SF4 formulation of the flow equations (Section 4.3.3).

Clearly, the interpolation can become quite complex with these extensions of the direct interpolation. To prevent excessive and cost ineffective work, Stueben's recommendation of truncating the interpolation is adopted, where all entries that are smaller than the largest by a filter factor, ξ_{tr}, are ignored, and all remaining entries re-scaled to conserve the row sum [6].

C-AMG will be applied to both coupled field formulations, PV2, and scalar field formulations SF4.

3.3. Velocity–pressure coupling in the CGA

The problem of velocity–pressure coupling has been a recurrent theme throughout the development of discrete approximations for incompressible fluid flow. The two established methods for ensuring good coupling, and hence stable discretizations, are the mixed (staggered/dual) grid approaches of finite-volume schemes and the mixed-interpolation approaches of the finite-element method. The SF4 scheme of Section 2.2 is a staggered/dual grid scheme and the PV2 scheme of Section 2.1 may also be viewed as a special form of dual grid scheme.

For AMG, it is not clear if the quality of the velocity–pressure coupling in the CGAs will automatically follow that for the fine grid. As already mentioned, this is because the equations for

DOI: 10.1002/fld

those grids are algebraically determined entirely from consideration of the intra-field couplings (off-diagonal entries of diagonal blocks of the system matrix) without any regard for the resulting inter-field couplings (entries in the off-diagonal blocks on the coarse-grid system matrix). Therefore, with a view to mitigating any possible degradation of the inter-field coupling during that process, we borrow the idea of mixed interpolation from the finite-element approach [26]. Thus, we allow for mixed-order interpolations in the formation of the inter-grid transfer operators for the application of AMG to the coupled vector–scalar scheme PV2. The notations SA-AMG(n_v, n_s) and C-AMG(n_v, n_s) will be adopted, where n_v and n_s are the nominal orders of the interpolations for the vector and scalar variables, respectively. The order is only 'nominal' because, as used in its normal sense, it cannot be guaranteed by using a purely algebraic coarsening process. Thus, in the case of SA-AMG(n_v, n_s), $n_v = 1$, $n_s = 0$, signifies the mixed interpolation resulting from a smoothed-aggregation for velocity and an unsmoothed-aggregation for pressure. Similarly, $n_v = 1, n_s = 1$, signifies the use of smoothed-aggregation for both velocity and pressure. In the case of C-AMG, $n_v = 1$, $n_s = 1$, also signifies an equal-order interpolation but in this case with standard interpolation being used for both velocity and pressure. The effectiveness of the strategy will be investigated by numerical experiments (Section 3).

In the case of the SF4 formulation, the coupling problem is avoided completely, since incompressibility is an intrinsic property of the scalar stream-function space and the CGAs are each represented by a single-block matrix equation. For consistency, notations SA-AMG(n_s) and C-AMG(n_s) will be used. For these fourth-order difference equations, the interpolation should, strictly, be higher than first order [27]. In this context the effectiveness of the Jacobi-relaxed interpolation will be tested by numerical experiment in Section 3.

3.4. Stabilized smoothing

The system matrices for both PV2 and SF4 formulations are not M-matrices. Even diagonal blocks of the coupled-block matrix are not M-matrices. This would normally compromise the stability of simple relaxation smoothing, and hence one of the two fundamental principles underpinning the AMG method as outlined above. To overcome this difficulty, a stabilizer in the form of a minimum residual (GCR) control harness is used to drive the smoother, which may then take the usual simple form of either Jacobi or Gauss–Seidel relaxation. This is particularly important for the coupled system where the multiple smoother sweeps are normally required for optimum efficiency.

In addition to the normal V-cycle smoothing schedule, provision is made for the more complex F-cycle schedule that provides for a greater investment of multigrid processing for the coarser grids. Algorithms may be found in Reference [9]. The notation $F(v_2, v_1)$ will be used to qualify the solver, for example, SA-AMG(1, 0)F(3, 0).

3.5. Acceleration

Recognizing that the CGAs in AMG may not always be sufficient, provision has been made for accelerators, GCR, GMRES, BICGSTAB, PCG, etc. Of these GCR has been chosen because a one-to-one correspondence between GCR iterations and AMG F-cycles is convenient for the comparison and presentation of results. The accelerator is only used where it improves efficiency. Where it is used, the qualifier GCR is appended in the notation, e.g. C-AMG(1)F(1, 0)GCR, for the case of GCR accelerated C-AMG, with the F(1, 0) cycle, when applied to the scalar field SF4 formulation of the problem.

DOI: 10.1002/fld

3.6. Preconditioning

The SF4 system matrix was scaled by binormalization using the method of Livne and Golub [28]. This is a type of preconditioning in which rows and columns are scaled in the Euclidean norm. The scaled matrix is better suited to AMG [29] and has improved numerical and stability properties.

4. NUMERICAL EXPERIMENTS

4.1. Test problem

To meet the requirements of multi-dimensional flow, the driven cavity is chosen as the test problem; just two spatial dimensions to maximize the range of available mesh bandwidth within the constraints of limited computing resources. A Reynolds number of 100 is chosen, high enough to provide an advection dominated flow, yet low enough not to compromise the lower range of available mesh resolution. The mesh resolving power, Q, needs to satisfy $Q>Re^{1/2}$ in order for the boundary layers to be resolved. Since the investigation is concerned with the scaling of performance with mesh size, the discrete approximation itself should not contain any avoidable mesh dependence. For this reason, upwind differencing of advection terms is not used; the false diffusion associated with it is of order uLQ^{-1}, where u is the magnitude of the flow and L a characteristic length scale for the problem. Hence, central differencing is adopted despite the fact that this makes the problem numerically more difficult, and despite the fact that this will incur a mesh-dependent, dispersive, truncation error which scales as Q^{-2}. However, the dispersion should not significantly distort the scaling results over the range of mesh sizes to be studied, except possibly for extremely coarse meshes, $Q\sim 10$, which will be avoided.

4.2. Meshes

As emphasized by Perot [21], methods based on unstructured, staggered/dual, grids have numerous attractive mathematical properties but high-order accuracy is not one of them. For this reason we do not pursue the investigation of highly distorted meshes where the accuracy of the SF4 discretization is expected to be poor. Thus, meshes expected to provide second-order accuracy are the main choice, for example, uniform meshes. However, meshes expected to give first-order accuracy are also examined. These include non-uniform meshes and random meshes, which involve moderate distortions of cells (Figure 1). This prudence in mesh choice is also motivated by the recognition that poor AMG convergence may sometimes be connected more with deficiencies in the discretization rather than with deficiencies in the AMG method itself [7, 10]. For SA-AMG(1, 0) solutions on meshes with extreme distortion see Reference [10].

4.2.1. Uniform meshes. These take the form of Delaunay triangulations. They are optimized by removing all nodes with low-order connectivity (connectivity, χ, being defined as the number of intersecting edges). Thus, nodes with $\chi<5$ are removed as are quintuplet pairs, giving connectivities for all nodes within the range, $5\leqslant\chi\leqslant 7$.

4.2.2. Random meshes. Random meshes are generated from uniform meshes by perturbing the nodal positions of the uniform meshes randomly within specified limits, e.g. 10 and 25%.

DOI: 10.1002/fld

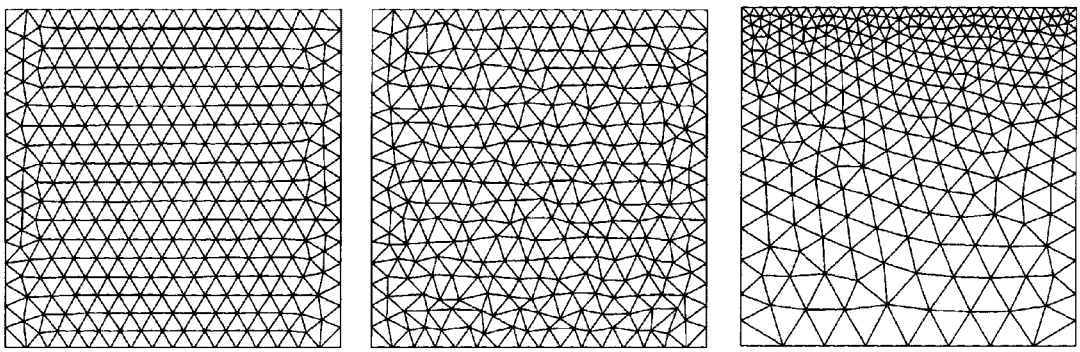

Figure 1. Examples of the three basic mesh types: uniform mesh, random mesh and smoothly graded non-uniform mesh. The examples are for the coarsest meshes used, mean bandwidth of $Q \sim 16$.

4.2.3. Non-uniform meshes. The non-uniform meshes are Delaunay triangulations smoothly graded with a four-fold refinement from the bottom to the top surface of the cavity. Again they are optimized so that connectivities fall within the range, $5 \leqslant \chi \leqslant 7$.

4.2.4. Mesh bandwidth. The scaling of solver performance is of particular importance in this investigation so some relevant measure of the mesh 'size' is required. To this end a nominal mesh bandwidth, Q, is defined to be synonymous with resolving power as

$$Q = \frac{L}{\delta}$$

for a characteristic nodal displacement (or element size), δ. For the non-uniform meshes and random meshes, an average value for δ is used to give a nominal average bandwidth.

4.3. Solver parameters

4.3.1. Smoothing. The GCR-controlled Jacobi smoother ($\nu_2 = 1$; $\nu_1 = 0$) was found to be the most efficient for the SF4 formulation. It has also been used for the PV2 formulation but in this case with the schedule ($\nu_2 = 3$; $\nu_1 = 0$).

4.3.2. Coarsening. For C-AMG, ξ_{st}, was set at 0.25 and ξ_{tr} at 0.05. However, when C-AMG was applied to the coupled PV2 formulation of the problem, some tuning of ξ_{st} was necessary. To obtain a more reliable convergence over the full range of mesh sizes, the following grid-dependent values, ξ_{st}^g, were used

$$\xi_{\mathrm{st}}^g = \xi_{\mathrm{st}} \left(\frac{1}{2}\right)^{g-1}; \quad 1 \leqslant g \leqslant G \tag{22}$$

g is the grid level and G the level of the coarsest grid. However, a few instances of stagnation were still observed, especially for non-uniform meshes.

SA-AMG was robust. No tuning was necessary. Performance was not sensitive to mesh uniformity: Nor was it sensitive to the precise values of either the strong coupling parameter, ξ_{st}, or the matrix truncation parameter, ξ_{tr}; they were set at $\xi_{\mathrm{st}} = \xi_{\mathrm{tr}} = 0.08$.

DOI: 10.1002/fld

4.3.3. Interpolation. For SA-AMG, the operator smoothing, where implemented, is applied to both restriction and prolongation in the form of a Jacobi smoothing with relaxation parameter 2/3.

For C-AMG Jacobi-relaxed interpolation (21), P_i is chosen to be the full set of coarse neighbours regardless of the strength of the connections to point, i, $P_i = C_i = N_i \cap C$.

4.4. Solver performance measures

4.4.1. Convergence. This investigation is primarily concerned with the performance of AMG as a linear solver for the iterative solution of non-linear equations. The convergence factor of interest is thus the residual reduction factors for AMG, not those for the non-linear solver. Non-linear solver convergence factors for this and other test problems for SA-AMG may be found in References [9, 22]. Here an overall average, ρ, is used to measure the quality of AMG performance,

$$\rho = \left[\prod_1^{\eta} \left\{ \left(\prod_1^{v} \rho^n \right)^{1/v} \right\} \right]^{1/\eta}, \quad \rho^n = \frac{\|\mathbf{r}^n\|_2}{\|\mathbf{r}^{n-1}\|_2}$$

where $\mathbf{r}^n$ is the residual for iteration, n, ρ^n is the corresponding reduction factor, v is the number of linear iterations to achieve a specified residual tolerance level and η is the number of non-linear iterations required to achieve a specified tolerance in the norm of the non-linear corrections. The tolerance for the linear solver is a 10^7 reduction in the residual norm. The tolerance for the non-linear solver is a 10^6 reduction in the change-norm. Of particular interest, in the assessment of the quality of the AMG solver, is the scaling of ρ with the mesh bandwidth Q.

4.4.2. Complexity. The overall scaling of AMG depends on the scaling of its complexity. Good convergence scaling would be of little value if it were gained at the cost of a mesh-dependent complexity. The algebraic complexity, C_A, is defined as the ratio of the total number of matrix entries for the complete set of, G, grids to the number of entries for the fine grid. Grid complexity, C_g, is defined as the ratio of the total number of equations for the complete set of, G, grids to the number of equations for the fine grid.

4.4.3. Efficiency. Attention will be focussed on the comparative efficiency and its scaling with problem size, N. SA-AMG(1, 0)F(3, 0) is chosen to provide the reference solutions. Computing times, τ, are discussed in terms of the scaling exponent, α, where it is assumed that $\tau \sim N^{\alpha}$.

4.5. Results for the coupled scalar–vector field: PV2

4.5.1. Convergence factors. The average convergence factors for the PV2 formulation on uniform unstructured meshes are shown in Figure 2. Those for the non-uniform mesh are given in Figure 3.

Consider first the equal-order interpolation results, SA-AMG(1, 1)F(3, 0) and C-AMG(1, 1) F(3, 0). The convergence factors are clearly Q-dependent, with step increases in ρ at mesh bandwidths of $Q = 32/40$ and at $Q = 96/100$, despite the fact that the order of the interpolations more than satisfies the requirement for a second-order discrete-difference system. The steps correspond to the incremental increases in the total number of coarse grids, G. This behaviour seems to be consistent with a reduction in the quality of the inter-field coupling at each grid coarsening (as suggested in Section 3.3). The larger the number of coarsening operations the poorer the quality of the overall representation of velocity–pressure coupling in the CGA. Observe that between the

DOI: 10.1002/fld

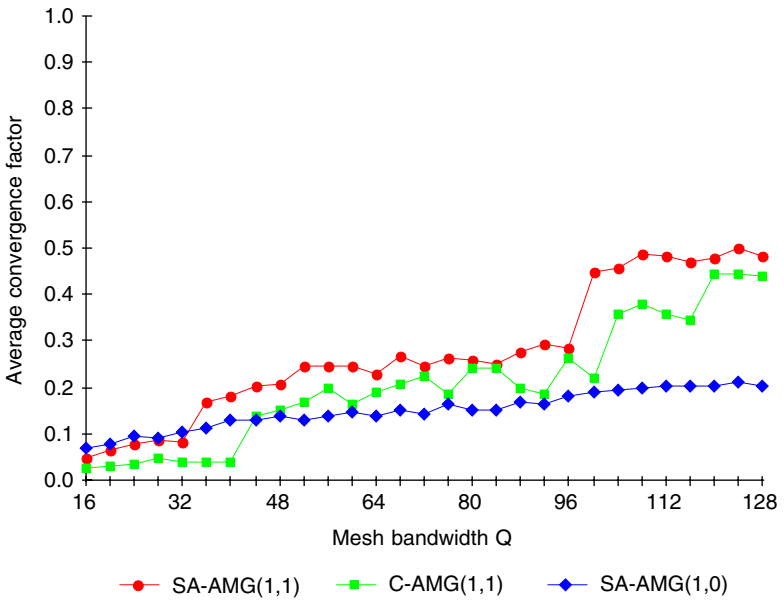

Figure 2. Scaling of average convergence factors for C-AMG and SA-AMG solvers; uniform meshes.

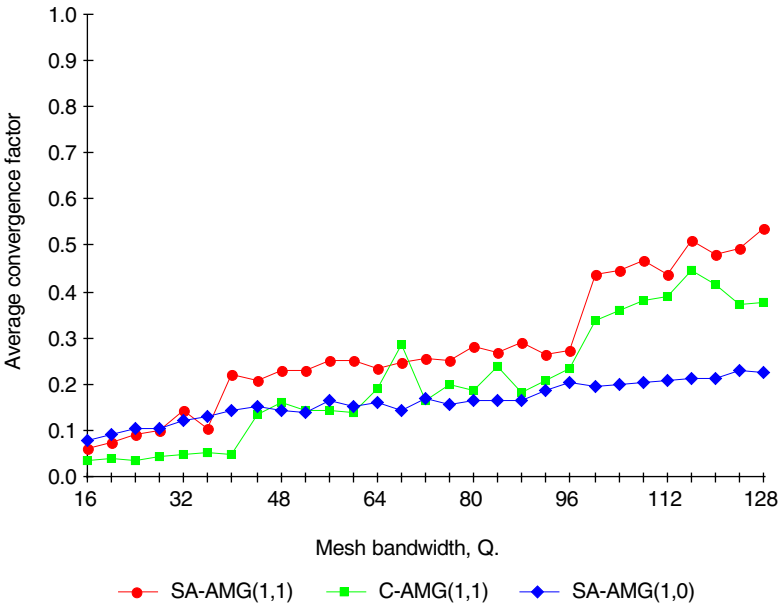

Figure 3. Scaling of average convergence factors for SA-AMG and C-AMG solvers; non-uniform meshes.

DOI: 10.1002/fld

steps the convergence factors do display, more or less, the normal Q-independent behaviour to be expected for first-order interpolations.

Note that the step behaviour is a characteristic of both solvers for both mesh types. Convergence factors for C-AMG tend to be slightly better than those for SA-AMG but are more erratic, especially for the non-uniform meshes (Figure 3). This is consistent with the fragility of the C-AMG convergence for this coupled field problem and the need for tuning of, ξ_{st}, as discussed in Section 4.3.2. However, even with the tuning, ξ_{st}^{g}, some instances of stagnation still occur, hence the high points in the scatter for C-AMG. No tuning was required for SA-AMG; convergence in this case was robust and insensitive to the precise value of ξ_{st}.

Consider now the mixed-order interpolations, SA-AMG(1, 0)F(3, 0) in Figures 2 and 3. In both cases these exhibit a much better, almost Q-independent, scaling without any obvious steps. This is consistent with the anticipated improvement in velocity–pressure coupling for mixed-order interpolations, as discussed in Section 3.3.

Mixed-order interpolation does indeed appear to maintain better inter-field coupling despite the fact that velocity and pressure systems are coarsened more or less independently without any consideration for inter-field coupling.

4.5.2. Algebraic complexity. In addition to giving a better scaling, SA-AMG(1, 0) has also the lowest complexity (Figure 4). That for C-AMG(1, 1) is up to 40% higher while that for SA-AMG(1, 1) is up to 10% higher. Moreover, for SA-AMG, there is little difference between the results for uniform and non-uniform meshes (both equal- and unequal-order interpolation). However, the complexity for C-AMG is about 5% larger on non-uniform meshes. Nevertheless, both solvers scale well. Complexity will not compromise the overall scaling of computing time with mesh size.

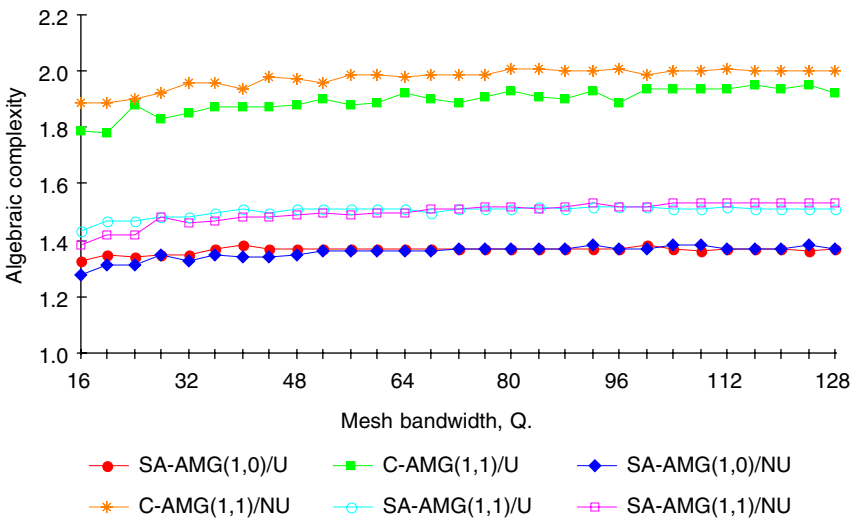

Figure 4. Scaling of algebraic complexity, C_A, for SA-AMG and C-AMG solvers, uniform (/U) and non-uniform (/NU) unstructured meshes.

DOI: 10.1002/fld

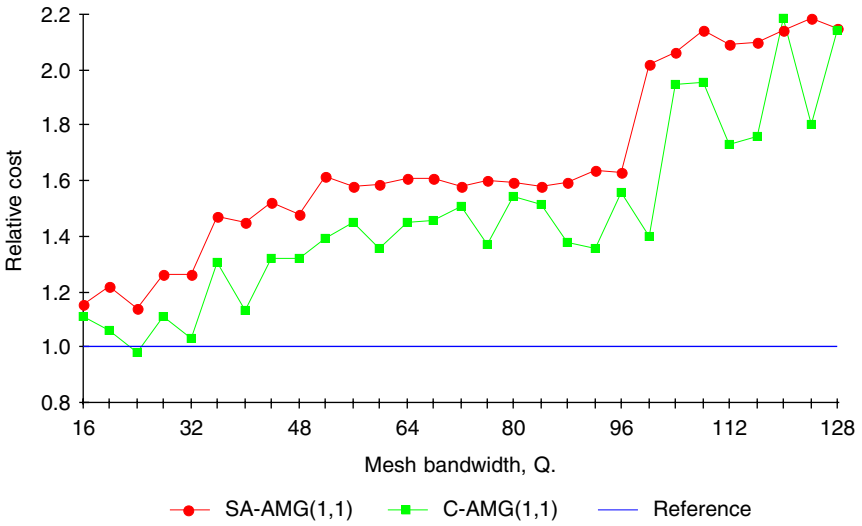

Figure 5. Scaling of the cost of SA-AMG(1, 1) and C-AMG(1, 1) solvers relative to SA-AMG(1, 0).

4.5.3. Relative efficiency. In view of the superiority of SA-AMG(1, 0) for both convergence factors and complexity, its overall performance is better than that for the other solvers both in terms of absolute efficiency and in terms of the scaling of computing time with problem size. This is clearly illustrated in Figure 5, where the cost of SA-AMG(1, 1) and C-AMG(1, 1) relative to that for SA-AMG(1, 0) is plotted as a function of mesh bandwidth. The form of the scaling obviously reflects that of the convergence factors. However, note that even for coarse meshes where C-AMG(1, 1) convergence factors are much better than those for SA-AMG(1, 0) the cost is no better (relative cost ~ 1.0) because of its higher complexity.

In conclusion, SA-AMG(1, 0), offers almost mesh-independent solutions for the coupled velocity–pressure formulation of the Navier–Stokes equations, PV2; the scaling exponent for computing time, $\alpha = 1.09$, is close to the optimum, $\alpha = 1.0$.

4.6. Results for the scalar field: SF4

For this formulation we have just one matrix equation (12) for the, discrete stream-function, **s**. Primitive variables, $\mathbf{U}, \mathbf{v}^c$ and **p** are recovered in post-processing steps using Equations (11) and (6). The cost of both this post-processing and the pre-processing required to form (12), is included in the following assessments. Although it is a single-block matrix equation, it presents a more difficult challenge for the AMG solvers since the matrix represents a fourth-order discrete-difference operator, whereas, the solvers above were formulated for just second-order difference equations. Strictly, as noted earlier, a higher order interpolation is required for at least one of the inter-grid transfer operators. Although there has been some research on SA-AMG methods for fourth-order, discrete-difference, equations [23], the solver produced was not based entirely on information contained in the system matrix. Research has intensified in recent years on the development of an adaptive SA-AMG, sometimes designated αSA [30], which is based entirely on information contained in the matrix, however, it is not clear at this stage if this will deliver

DOI: 10.1002/fld

a practical linear solver for these non-linear fluid flow problems. Here, therefore, the above SA-AMG and C-AMG solvers are applied to the problem in the knowledge that performance is likely to fall short of the ideal.

4.6.1. Convergence factors. Convergence factors for C-AMG(1)F(1, 0) and SA-AMG(1)F(3, 0) GCR solvers for uniform, unstructured, meshes are presented in Figure 6.

Where applied, the smoothing, F(3, 0), and the accelerator, GCR, have been used to optimize performance. The scaling for SA-AMG(1)F(3, 0)GCR is mesh-dependent, as expected. Note, however, that the scaling of C-AMG(1)F(1, 0) is better. Within the point-to-point scatter, the convergence is independent of the mesh bandwidth, Q. This, somewhat surprising, optimum scaling is only realized, however, when the Jacobi-relaxed interpolation is used. If direct interpolation or standard interpolation is used, then the convergence factors are mesh dependent, just like those for SA-AMG(1)F(3, 0)GCR. This optimum scaling is also lost on non-uniform meshes; see Figure 7, which shows the scaling of the average convergence factors for C-AMG(1)F(1, 0)GCR on random and smoothly graded non-uniform meshes (results for uniform mesh are included for comparison).

For the random meshes, the strength of the mesh dependence is directly related to the degree to which the mesh is perturbed from the uniform (note that angles in some elements exceed 120° on the most perturbed mesh). Even for the smoothly graded, non-uniform (Delaunay) mesh, where the 4:1 refinement is spread evenly across the entire domain, the mesh dependence is still significant. It would seem, therefore, that the Jacobi-relaxed interpolation is just sufficient to deliver optimum scaling for a uniform mesh but is unable to provide the same quality of CGA if the meshes depart even a small amount from that uniformity.

Note that it is only for such uniform meshes that the SF4 formulation is able to deliver a second-order accurate solution. For non-uniform meshes the accuracy degrades to first order, whereas the accuracy for the PV2 formulation is second-order for both uniform and non-uniform meshes.

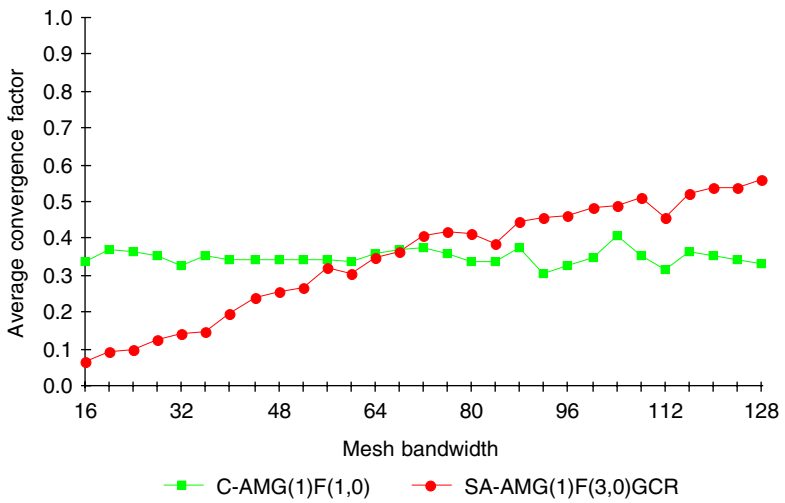

Figure 6. The scaling of ‘average convergence factors for SA-AMG(1)F(3, 0)GCR and C-AMG(1)F(1, 0).

DOI: 10.1002/fld

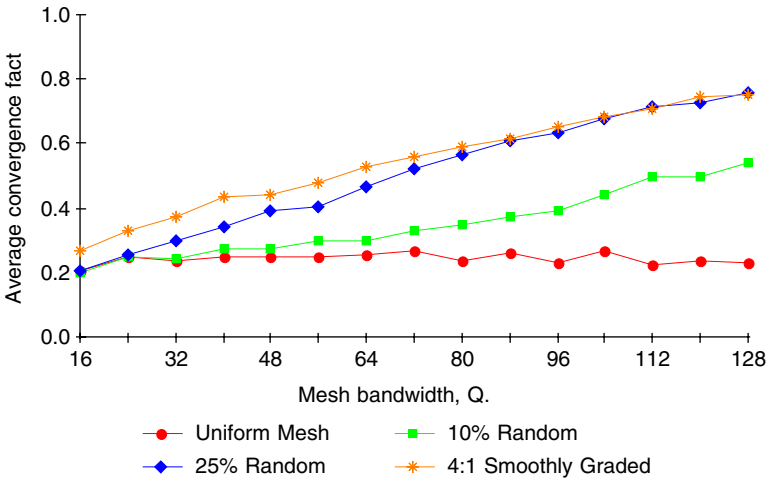

Figure 7. The scaling of average convergence factors for C-AMG(1)F(1, 0)GCR.

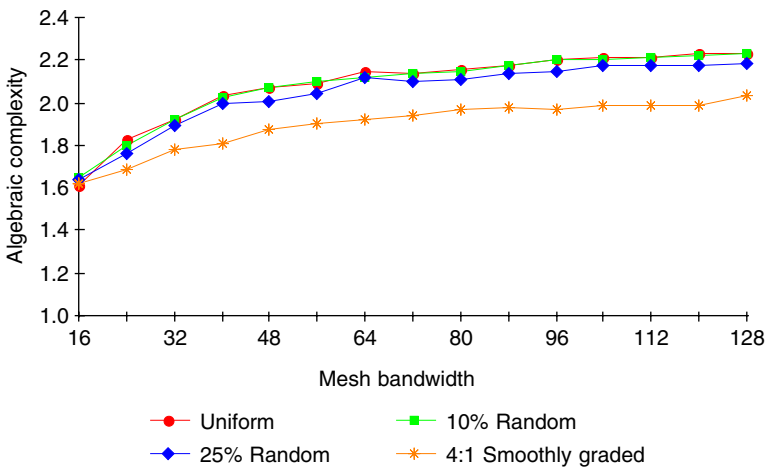

Figure 8. The scaling of algebraic complexity, C_A, for the C-AMG-based solvers.

4.6.2. Algebraic complexity. The algebraic complexities for C-AMG(1) show some mesh dependence at low mesh bandwidths, but improve as bandwidth increases (Figure 8, compare with results for the PV2 formulation, Figure 4).

The smoothly graded, non-uniform mesh has a slightly lower complexity than those for the uniform and random meshes but the asymptotic values are in the same range as for C-AMG on the PV2 formulation. All are much larger than the complexities for SA-AMG.

DOI: 10.1002/fld

4.6.3. Comparative efficiency. The scaling of the relative costs of C-AMG(1)F(1, 0) and of C-AMG(1)F(1, 0)GCR, for uniform and non-uniform meshes, respectively, are shown in Figure 9.

Clearly, they are more costly than the reference solutions by a factor of between 1.4 and 2.2 over the range of mesh bandwidth. This may seem somewhat surprising in view of the fact that the SF4 system is a single-block matrix, whereas the coupled PV2 system contains 7 non-zero blocks (10 non-zero blocks in 3D). However, it is to be remembered that assembling the SF4 block requires the successive transformation of two coupled multi-block systems. These are the cell-based, 7-block system for $\mathbf{v}^c$ and $\mathbf{p}$, each approximately twice as large, and the cell-face based 4-block system for $\mathbf{U}$ and $\mathbf{p}$, the U-blocks being roughly three times as large. Also, the complexity of C-AMG is larger, by about 50%, than that for SA-AMG. These additional SF4 costs only increase linearly with problem size, so they do not compromise the overall solver scaling. In fact, they weaken the overall effective mesh-dependence, since solution phase costs are then a smaller fraction of the total cost. The larger complexity of C-AMG is also only weakly mesh dependent, so this too effectively reduces the mesh dependence. It is for this reason that the scaling for non-uniform meshes, case 'B' in Figure 9, seems better than expected (bearing in mind the scaling for the convergence factors, Figure 7). For a 64-fold increase in problem size there is only a 60% increase in relative cost.

So even for non-uniform meshes, C-AMG (with Jacobi-relaxed interpolation) has relatively weak mesh dependence. The scaling exponent, for this range of mesh bandwidth, is $\alpha = 1.2$. C-AMG can therefore be used as a practical solver for such first-order accurate SF4 formulations.

For uniform meshes on the other hand, where the SF4 formulation is second-order accurate, case 'A' in Figure 9, the relative cost of C-AMG is mesh independent on average. This suggests that in terms of the absolute cost, the mesh dependence is similar to that for the reference. The scaling exponent, α, is estimated to be $\alpha = 1.06$ (if the coarser grid results are discounted) which

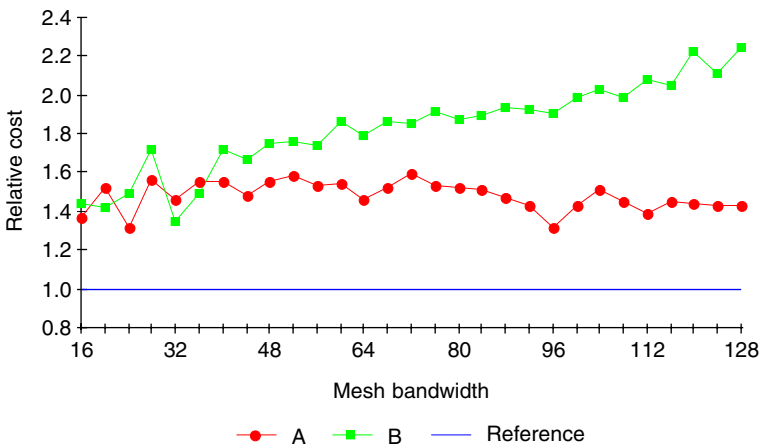

Figure 9. The scaling of the relative costs: Case 'A': C-AMG(1)F(1, 0); uniform unstructured meshes. Case 'B': C-AMG(1)F(1, 0)GCR; smoothly graded, non-uniform, unstructured meshes. The reference costs are for SA-AMG(1, 0)F(3, 0) applied to the PV2 formulation on uniform and non-uniform unstructured meshes, respectively.

DOI: 10.1002/fld

compares with $\alpha = 1.09$ for SA-AMG(1, 0). Where SA-AMG(1, 0) is weakly mesh dependent in its convergence (Figure 2), C-AMG(1) is weakly mesh dependent in its complexity (Figure 8).

4.7. General comments

4.7.1. Coupled SA-AMG solver for the PV2 formulation. The two important factors in achieving almost Q-independent convergence for the coupled SA-AMG solver are the stabilization of smoothing using a GCR control harness and the maintenance of good inter-field coupling in the CGAs using mixed-order interpolation. This then makes the overall performance of the solver similar to that reported for single-field problems [9].

4.7.2. C-AMG solver for the SF4 formulation. The point-to-point variations in the C-AMG results for the SF4 formulation on (quasi) uniform unstructured meshes (Figures 6 and 9) are partly due to the sensitivity of the convergence to mesh uniformity. They may be reduced by strictly enforcing uniformity of cell geometry, e.g. by triangulating a Cartesian mesh. This is illustrated in Figure 10, which shows that in this case the scatter is much reduced and comparable with that for the SA-AMG results for PV2.

It should be remembered that this particular implementation of the SF4 formulation represents a difficult challenge for iterative solvers. Single-level solvers, with simple preconditioning, struggle to deliver convergence, especially on the larger meshes. To illustrate this (and to put the mesh dependencies into perspective), Figure 11 shows the relative cost of Jacobi-preconditioned GCR and BiCGSTAB-based solvers for the SF4 system on uniform unstructured meshes. Since in both cases, the number of linear-solver iterations increases roughly in proportion to $\sim Q^2$, the overall relative cost scales as $\sim Q^\beta$, $\beta \geqslant 2$. In the case of BiCGSTAB, exponent, β, is not much larger than 2. However, in the case of GCR, the dimension of the Krylov subspace increases with the

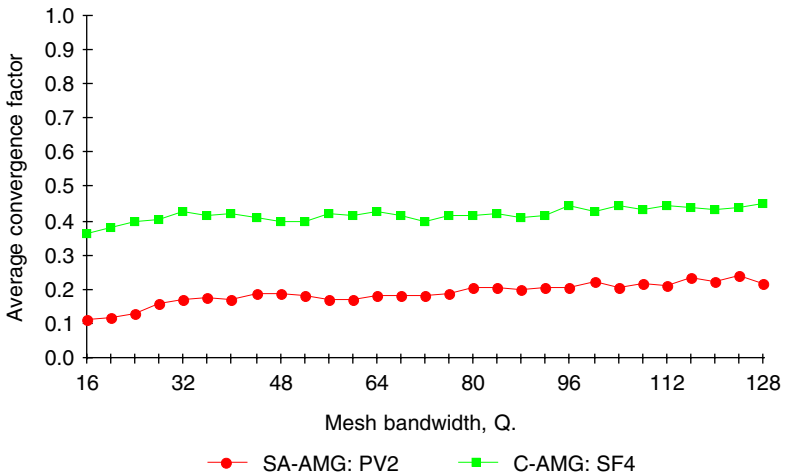

Figure 10. The scaling of average convergence factors for solvers SA-AMG(1, 0)F(3, 0) and C-AMG(1)F(1, 0) applied to the PV2 and SF4 formulations of the driven cavity test problem on uniform Cartesian meshes.

DOI: 10.1002/fld

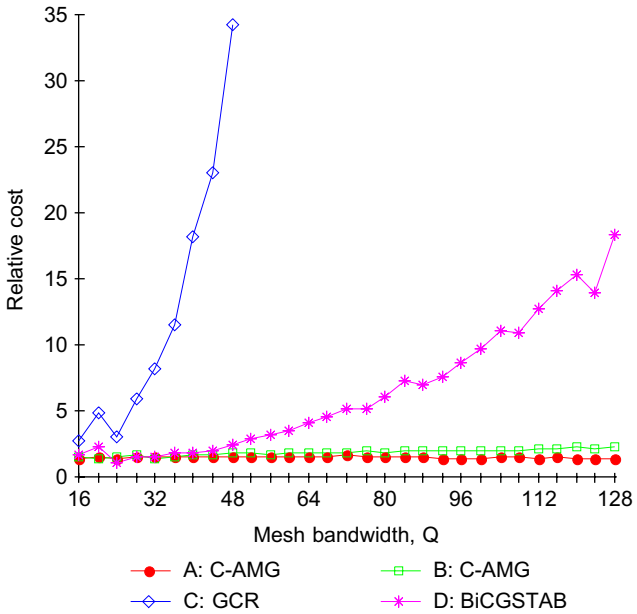

Figure 11. The scaling of relative costs: Case 'A': C-AMG(1)F(1, 0); uniform unstructured meshes. Case 'B': C-AMG(1)F(1, 0)GCR; smoothly graded, non-uniform, unstructured meshes. Case 'C': GCR; uniform unstructured meshes. Case 'D': BiCGSTAB; uniform unstructured meshes. The reference is SA-AMG(1, 0)F(3, 0) for PV2 on uniform unstructured meshes.

iteration count so the exponent is closer to 4. The Q dependence of the C-AMG-based solvers is clearly weak in comparison, for both uniform and non-uniform meshes.

Note in Figure 11 that the mesh dependence for GCR and BiCGSTAB solvers appears less strong at the lower mesh bandwidths. This is because the relative cost of the set-up phase, which is constant, is here a more significant fraction of the total cost. Furthermore, dispersion errors, associated with the central differencing of advection terms, are larger on coarser meshes and these slow up the convergence of the outer, non-linear, iterations. The latter effect is most pronounced at $Q = 20$ where the total relative cost is actually higher than at $Q = 24$. The performance of both the C-AMG-based solvers seems less sensitive to the effects of this dispersion error.

Krylov methods can be made more viable for SF4 formulations, by improving preconditioning, and by removing the implicit treatment of advection and introducing under-relaxation in the form of finite time steps, as in Reference [11]. The latter modifications make the iteration matrix symmetric and diagonally dominant and also permit PCG solution options. Of course, they would also make for easier AMG solutions.

To improve the scaling of C-AMG(1) for non-uniform meshes, an improved interpolation is required. Some of the current research activities on adaptive AMG [24] may provide this. The cost penalty of the adaptive phase would need to be mesh independent, of course. Also, for the benefits to be manifest within the range of accessible mesh bandwidth the additional cost must not be too large. In the mean time, the above results show that the current approach gives solutions at a cost that is not prohibitive.

DOI: 10.1002/fld

4.7.3. Tolerances. Convergence criteria chosen are not representative of those that might be used in real practical applications. For example, a 10^7 reduction in the residual norm for each linear approximation is probably excessive. It has been adopted here simply to ensure that a stagnated linear-solver convergence would not pass unnoticed. Less stringent reductions (of say 10^2–10^3) usually suffice in practice to ensure an efficient, overall, convergence to any desired level (above that dictated by finite machine arithmetic).

4.7.4. Other test cases. Results presented are for a test case chosen purely for presentational purposes. The SA-AMG mixed-interpolation approach has in fact been applied to many test problems, a wide range of flow conditions (from the Stokes limit to $Re = 3200$) and for a wide range of meshes, some with extreme distortion (highly stretched grids with element aspect ratios exceeding 300). In fact, results published in a previous paper, concerned with other matters, were obtained with SA-AMG(1, 0) and the PV2 formulation [10].

4.7.5. Tuning. The requirement for tuning in the case of the C-AMG(1, 1) solver when applied to the PV2 formulation must not be interpreted as a general requirement for solvers in the general classification of classical AMG, or even for this particular solver when applied to other problems. For example, no tuning was necessary for C-AMG(1) when applied to the SF4 formulation.

5. CONCLUSIONS

Two practical, AMG based, solution methods have been developed for incompressible fluid flow, one for a second-order, discrete-difference formulation of the Navier–Stokes equations in primitive-variables and one for a fourth-order, discrete-difference formulation in a single scalar-field variable.

For the second-order formulation, PV2, on both uniform and non-uniform unstructured meshes, a smoothed-aggregation-based AMG solver provides solutions with an almost optimum mesh-independent convergence. A GCR control harness is used to stabilize the smoothing and a mixed-order interpolation is used to ensure good velocity–pressure coupling in the CGA.

For the fourth-order formulation, SF4, on uniform unstructured meshes, the classical AMG-based solver (with a Jacobi-relaxed interpolation) provides solutions with convergence rates independent of mesh bandwidth and at an overall cost that is only weakly mesh dependent. For non-uniform meshes the convergence of this solver becomes mesh dependent, but the overall solution cost increases relatively slowly with mesh bandwidth.

ACKNOWLEDGEMENTS

I am grateful to Steve McCormick, Blair Perot, John Ruge and Klaus Stueben for reading this paper and making helpful comments.

REFERENCES

1. Brandt A, McCormick SF, Ruge JW. Algebraic multigrid (AMG) for automatic multigrid solutions with application to geodetic computations. *Report*, Inst. For Computational Studies, Fort Collins, CO, October 1982.
2. Brandt A. Algebraic multigrid (AMG) for sparse matrix equations. In *Sparsity and its Applications*, Evans DJ (ed.). Cambridge University Press: Cambridge, 1984.

DOI: 10.1002/fld

3. Stueben K. Algebraic multigrid (AMG): experiences and comparisons. *Applied Mathematics and Computation* 1983; **13**:419–452.
4. Ruge JW, Stueben K. Algebraic multigrid (AMG). In *Multigrid Methods*, McCormick SF (ed.). Frontiers in Applied Mathematics, vol. 3. SIAM: Philadelphia, PA, 1987; 73–180.
5. Stueben K. A review of algebraic multigrid. German National Centre for Information Technology (GMD). *GMD-Report 69*, November 1999.
6. Stueben K. An introduction to algebraic multigrid (Appendix A). In *Multigrid*, Trottenberg U, Oosterlee C, Schuller A (eds). Academic Press: London, 2001.
7. Cleary AJ, Falgout RD, Henson VE, Jones JE, Manteuffel TA, McCormick SF, Miranda GN, Ruge JW. Robustness and scalability of algebraic multigrid. *SIAM Journal on Scientific Computing* 2000; **21**:1886–1908.
8. Brannick J, Brezina M, MacLachlan S, Manteuffel T, McCormick S. An energy based AMG coarsening strategy. *Numerical Linear Algebra with Applications* 2005; **13**:1–16.
9. Webster R. Performance of algebraic multigrid solvers based on unsmoothed and smoothed aggregation schemes. *International Journal for Numerical Methods in Fluids* 2001; **36**:743–772.
10. Webster R. Stability of Navier–Stokes discretizations on collocated meshes of high anisotropy and the performance of algebraic multigrid solvers. *International Journal for Numerical Methods in Fluids* 2006 Published online in Wiley InterScience (www.interscience.wiley.com). DOI 10.1002/fld1181.
11. Chang W, Giraldo F, Perot B. Analysis of an exact fractional step method. *Journal of Computational Physics* 2002; **179**:1–17.
12. Harlow FH, Welch JE. Numerical calculations of time dependent viscous incompressible flow of fluid with a free surface. *Physics of Fluids* 1965; **8**(12):2182–2189.
13. Lilly DK. On the computational stability of numerical solutions of time dependent non-linear geophysical fluid dynamics problems. *Monthly Weather Review* 1965; **93**:11–25.
14. Hall CA, Cavendish JC, Frey WH. The dual variable method for solving fluid flow difference equations on Delaunay triangulations. *Computers and Fluids* 1991; **20**(2):145–164.
15. Nicolaides RA. The covolume approach to computing incompressible flow. In *Algorithmic Trends in Computational Fluid Dynamics*, Hussaini MY, Kumar A, Salas MD (eds). Springer: Berlin/New York, 1993.
16. Hyman JM, Shashkov M. Natural discretizations for the divergence, gradient and curl on logically rectangular grids. *Computers and Mathematics with Applications* 1997; **33**(4):81–104.
17. Hyman JM, Shashkov M. Adjoint operators for the natural discretizations of the divergence, gradient and curl on logically rectangular grids. *Applied Numerical Mathematics* 1997; **25**:413–442.
18. Hyman JM, Shashkov M. The orthogonal decomposition theorems for mimetic finite difference methods. *SIAM Journal on Numerical Analysis* 1999; **36**(3):788–818.
19. Shashkov M. *Conservative Finite-Difference Methods on General Grids*. CRC Press: Boca Raton/London/New York/Washington D.C., 1996.
20. Hyman JM, Shashkov M. Mimetic discretizations of Maxwells equations. *Journal of Computational Physics* 1999; **151**:881–909.
21. Perot B. Conservation properties of unstructured staggered mesh schemes. *Journal of Computational Physics* 2000; **159**:58–89.
22. Webster R. An algebraic multigrid solver for Navier–Stokes problems in the discrete second-order approximation. *International Journal for Numerical Methods in Fluids* 1996; **22**:1103–1123.
23. Vanek P, Mandel J, Brezina M. Algebraic multigrid by smoothed aggregation for second and fourth-order elliptic problems. *Computing* 1996; **56**:179–196.
24. Brezina M, Falgout R, MacLachlan S, Manteuffel T, McCormick S, Ruge J. Adaptive algebraic multigrid. *SIAM Journal on Scientific Computing* 2006; **27**:1261–1286.
25. Briggs WL, Henson VE, McCormick SF. *A Multigrid Tutorial* (2nd edn). SIAM Books: Philadelphia, 2000.
26. Hood P, Taylor C. Navier–Stokes equations using mixed interpolation. In *Finite-Element Methods in Flow Problems*, Oden JT *et al.* (eds). UAH Press: University of Alabama, Huntsville, AL, 1974; 57–66.
27. Hemker PW. On the order of prolongations and restrictions in multigrid procedures. *Journal of Computer Applications and Mathematics* 1990; **32**:423–429.
28. Livne OE, Golub GH. Scaling by binormalisation. Computer Science Dept. Report SCCM-03-12, SCCM, Stanford, 2003.
29. Livne OE. Coarsening by compatible relaxation. *Numerical Linear Algebra with Applications* 2004; **11**:205–227.
30. Brezina M, Falgout R, MacLachlan S, Manteuffel T, McCormick S, Ruge J. Adaptive smoothed aggregation (αSA). *SIAM Journal on Scientific Computing* 2004; **25**:1896–1920.