An AMG strategy for efficient solution of free-surface flows

A.G.B. Mowat[a], W.J. van den Bergh[b], A.G. Malan[a,*], D.N. Wilke[b]

[a] *Department of Mechanical Engineering, University of Cape Town, Cape Town 8000, South Africa*
[b] *Department of Mechanical and Aeronautical Engineering, University of Pretoria, Pretoria 0002, South Africa*

Abstract

An area of great interest in current computational fluid dynamics research is that of Free-Surface Modelling (FSM). Semi-implicit pressure based FSM flow solvers typically involve the solution of a pressure correction equation. The latter being computationally intensive, this work involves the implementation and enhancement of an Algebraic Multigrid (AMG) method for its solution. All AMG components were implemented in a manner which ensures linear computational scalability and matrix-free storage. In addition, a so-called Freeze method was developed to address the computational overhead resulting from the dynamically changing coefficient matrix. The latter involves periodic AMG setup steps in a manner that results in a robust and efficient black-box solver. The developed technology was evaluated in two- and three-dimensions via application to a dam-break test case. AMG performance was assessed via comparison of CPU cost to that of several other competitive sparse solvers. The standard AMG implementation proved inferior to other methods in three-dimensions, while the developed Freeze version achieved significant speed-ups and proved to be superior through-out.

Keywords: Algebraic Multigrid, Matrix-free, Pressure-Poisson, Free-Surface Modelling

*Corresponding author
Email address: arnaud.malan@uct.ac.za (A.G. Malan)

1. INTRODUCTION

One of the most challenging and relevant areas of study in modern computational fluid dynamics is that of free surface modelling (FSM). Efficient modelling of free surface flows is having a profound impact on many fields of engineering, as evidenced by the variety of problems analysed. Examples include fuel sloshing in tanks, breaking waves, flow around ship hulls and casting. The accurate and efficient modelling of these problems, however, has proven to be challenging due to the large density and viscosity jumps across the interface between two phases. Various methods exist to capture this behaviour, the most prevalent of which include volume of fluid (VoF) methods [1] and level set methods [2]. Regardless, high performance CFD tools are required for industrially usable FSM simulations.

Elemental$^{\text{TM}}$is one such tool, employing a fractional step pressure prediction solution method combined with CICSAM [3] VoF interface tracking scheme. The fractional step method consists of three distinct steps [4, 5], one of which involves obtaining a pressure correction value. In the case of the implicit pressure formulation, the latter involves the solution of a large, sparse and asymmetric linear system of equations. At present, this is effected by means of a preconditioned Generalised Minimal Residual (GMRES) solver [6, 7, 8]. Though this solver was demonstrated to be exceptionally efficient it was found to consume up to 95% computational time. Thus, the motivation for this work was to employ an alternative solution method in order to improve on solution times and reduce the contribution of the pressure correction solution step.

From the literature, a number of advanced sparse linear solvers have been applied to FSM for the purpose of solving the pressure correction equation. These include successive over-relaxation [9], preconditioned GMRES [10, 8], various conjugate-gradient type methods [11, 12, 13, 14, 15] and geometric multigrid methods [16, 17, 18]. Other attempts at solution acceleration include improved initial condition prediction [19] and algebraic multigrid (AMG) methods [20]. Despite its desirable qualities of being fast and matrix-free, AMG is never cited

as a preferred sparse solver for FSM simulations. This is expected to be due in part to computational overheads specific to FSM problems. These are a result of repeated setup costs due to the dynamic nature of the pressure correction coefficient matrix.

In an attempt to benefit from the computational efficiency of AMG while reducing the effect of repeated setup steps, this work is concerned with the development and implementation of an augmented AMG solver. With focus on modularity, a classical Algebraic Multigrid (C-AMG) [21] black-box solver was developed using an object oriented methodology [22]. Priority was given to obtaining optimal $O(N)$ algorithm complexity. The setup cost was addressed by extending the basic solver to a so-called Freeze-AMG (F-AMG) method, which alleviates repeated setup steps by maintaining solution structures over multiple time-steps in a robust and automatic fashion. Additionally, the Freeze method was extended to only perform partial setup phases as needed.

Following successful implementation, rigorous evaluation of the solver followed. This consisted of application to violent free-surface flows and comparing solution times using F-AMG and C-AMG to that of a range of pre-conditioned matrix-free sparse solvers *viz.* Conjugate Gradient (CG) , Bi-Conjugate Gradient Stabilised (BiCGSTAB) and GMRES. The latter solvers were purpose built for this work. Structured and unstructured meshes were employed in two- and three-dimensions. Performance evaluation metrics included computational time scaling as a function of mesh topology and CPU time as a function of simulated time.

2. THE PRESSURE CORRECTION EQUATION

2.1. Governing Equations

In many free-surface simulations, the set of governing equations is the well known incompressible Navier-Stokes. Combined with an interface tracking equation, arising from the VoF method, these can be written for a two fluid system

in terms of locally averaged quantities as:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^j}{\partial x_j} + \frac{\partial \mathbf{H}^j}{\partial x_j} - \frac{\partial \mathbf{G}^j}{\partial x_j} = \mathbf{S} \tag{1}$$

where $\mathbf{S}$ is a vector of body forces. The flow and flux variables are

$$\mathbf{U} = \begin{pmatrix} \rho_m u_1 \\ \rho_m u_2 \\ \rho_m u_3 \\ 0 \\ \alpha \end{pmatrix}, \ \mathbf{F}^j = \begin{pmatrix} \rho_m u_1 u_j \\ \rho_m u_2 u_j \\ \rho_m u_3 u_j \\ u_j \\ \alpha u_j \end{pmatrix}, \ \mathbf{H}^j = \begin{pmatrix} p\delta_{1j} \\ p\delta_{2j} \\ p\delta_{3j} \\ 0 \\ 0 \end{pmatrix}, \ \mathbf{G}^j = \begin{pmatrix} \sigma_{1j} \\ \sigma_{2j} \\ \sigma_{3j} \\ 0 \\ 0 \end{pmatrix}, \tag{2}$$

where δ_{ij} is the Kronecker delta and the viscous stress for a Newtonian fluid is given by

$$\sigma_{ij} = \mu_m \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right). \tag{3}$$

In the above equations, t and x_j denote time and Cartesian coordinate j respectively, ρ_m is the local volume averaged density, $\mathbf{u}$ is the velocity field, p is the pressure and μ_m the mean viscosity. The coefficient α is the volume fraction of one of the fluids.

2.2. Solution Procedure

In order to solve the governing equation set, *Elemental*$^{\text{TM}}$ employs a fractional step method [23, 24]. Here the velocity u_i and the pressure p at the current time-step n are employed to obtain values at the next time-step $n+1$. In this work, we consider the semi-implicit method which commences by calculating an intermediate velocity variable as:

$$\frac{\Delta U_i^*}{\Delta t} = - \left. \frac{\partial F^{ij}}{\partial x_j} \right|^n + \left. \frac{\partial G^{ij}}{\partial x_j} \right|^n + \left. S_i \right|^n, \tag{4}$$

for $i = [1, 2, 3]$, where Δt denotes the physical time-step size, $\Delta t = t^{n+1} - t^n$. Next, the pressure correction equation is constructed in an implicit fashion as

$$0 = \Delta t \frac{\partial}{\partial x_i} \left(\frac{1}{\rho_m} \frac{\Delta U_i^*}{\Delta t} - \frac{1}{\rho_m} \frac{\partial p^{n+1}}{\partial x_i} \right) + \left. \frac{\partial u_i}{\partial x_i} \right|^n, \tag{5}$$

which may be rewritten as:

$$\frac{\partial}{\partial x_i}\left(\frac{1}{\rho_m}\frac{\partial p^{n+1}}{\partial x_i}\right) = \frac{\partial}{\partial x_i}\left(\frac{1}{\rho_m}\frac{\Delta U_i^*}{\Delta t}\right) + \frac{1}{\Delta t}\frac{\partial u_i}{\partial x_i}\bigg|^n . \tag{6}$$

The pressure correction may now be solved implicitly from this equation (as described below).

In the third and final step, the calculated pressure is used to calculate the velocity at the next time-step as:

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \frac{\Delta U_i^*}{\Delta t} + \frac{\partial H^{ij}}{\partial x_j}\bigg|^{n+1} . \tag{7}$$

The interface position is updated using:

$$\frac{\alpha^{n+1} - \alpha^n}{\Delta t} = \frac{\partial\left(\alpha u_i\right)}{\partial x_i}\bigg|^n , \tag{8}$$

where the nomenclature is as defined previously.

2.3. Spatial Discretisation and Equation Construction

Commencing with pressure solution, Equation (6) is written as:

$$K(p^{n+1}) = \frac{\partial}{\partial x_i}\left(\frac{1}{\rho_m^n}\frac{\partial p^{n+1}}{\partial x_i}\right) = \frac{\partial}{\partial x_i}\left(\frac{1}{\rho_m}\frac{\Delta U_i^*}{\Delta t}\right) + \frac{1}{\Delta t}\frac{\partial u_i}{\partial x_i}\bigg|^n , \tag{9}$$

which is read to mean that some K exists which is linear in p^{n+1}, with $K = f(p_m, \rho_m)|^n$, where the superscript n refers to the current time step. The equation is now linearised around n as:

$$K\left(p^{n+1}\right) = K\left(p^n\right) + \frac{\partial K}{\partial p}\bigg|^n \Delta p = \frac{\partial}{\partial x_i}\left(\frac{1}{\rho_m}\frac{\Delta U_i^*}{\Delta t}\right) + \frac{1}{\Delta t}\frac{\partial u_i}{\partial x_i}\bigg|^n \tag{10}$$

$$\Rightarrow \frac{\partial K}{\partial p}\bigg|^n \Delta p = \frac{\partial}{\partial x_i}\left(\frac{1}{\rho_m}\frac{\Delta U_i^*}{\Delta t}\right) + \frac{1}{\Delta t}\frac{\partial u_i}{\partial x_i}\bigg|^n - K(p^n), \tag{11}$$

where $\Delta p = (p^{n+1} - p^n)$ is the pressure correction sought.

The manipulated pressure correction equation is now discretised using an edge-based approach similar to that in [6], resulting in the following system of linear algebraic equations to be solved:

$$\mathbf{A}\Delta\mathbf{p} = \mathbf{b}, \tag{12}$$

where $\mathbf{A}$ is now a sparse, asymmetric coefficient matrix, $\Delta\mathbf{p}$ the pressure correction and $\mathbf{b}$ the right hand side of the equation system. Importantly, the matrix $\mathbf{A}$ varies as a function of time as the surface interface evolves, which strongly affects ρ_m as density varies by a factor 1000 due interface motion.

3. DEVELOPED SOLVER TECHNOLOGY

3.1. Object-Oriented AMG Solver

So called C-AMG [21], has proven to be an effective and robust method for solving linear systems. In this work, it was necessary to implement an AMG solver as a stand alone black-box subroutine into *Elemental*$^{\text{TM}}$. To effect this a self-contained AMG solver object was developed which is accessed whenever the pressure correction equation is to be solved. The object accepts a sparse system matrix in compressed row format, Equation (12), constructs the algebraic multigrid hierarchy according to the C-AMG principles found in [25], and returns the pressure correction solution. This is done repeatedly for successive time-steps as shown schematically in Figure 3.

A basic outline of C-AMG [25] is given as follows. For convenience the grid points associated with $\mathbf{A}$ are denoted as $\Omega = \{1, 2, 3, ..., N\}$, where N denotes the grid size. Superscripts denote level number, such that $\mathbf{A} = \mathbf{A}^1$ and $\Omega = \Omega^1$ represent the finest grid. Prior to the C-AMG solver a *setup phase* is required to decompose $\mathbf{A}$ into consecutively coarser grids ($\Omega^1 \supset \Omega^2 \supset \Omega^3 \supset ... \supset \Omega^M$) which result in a series of coarse $\mathbf{A}^k$, interpolation operators I_{k+1}^k and restriction operators I_k^{k+1}.

Setup Phase

1. Set $k = 1$, where k denotes the grid level (fine to coarse).

2. Partition Ω^k into unique sets of coarse (C^k) and fine (F^k) grid points.
   (a) Set $\Omega^{k+1} = C^k$.
3. Define interpolation I_{k+1}^k and restriction $I_k^{k+1} = \left(I_{k+1}^k\right)^T$ operators.
4. Construct next level $\mathbf{A}^{k+1} = I_k^{k+1} A^k I_{k+1}^k$ (Galerkin condition).
5. If the size of Ω^{k+1} is smaller than a stopping size then set maximum level, $M = k + 1$. Otherwise set $k = k + 1$ and return to step 2.

As per the above, there are two critical steps in the *setup phase*, that being the selection of coarse and fine grid points and the construction of the transfer operators. C-AMG uses the classic coarsening algorithm by Ruge and Stüben [21]. The aforementioned algorithm starts by determining the strength of coupling between grid points. Point i is said to strongly depend on j if,

$$-a_{ij} \geq \theta \max_{k \neq i} \{-a_{ik}\} \tag{13}$$

where $0 < \theta < 1$ denotes the coarsening threshold and a_{ij} the entry of $\mathbf{A}$ (row i and column j). Point j is also said to strongly influence i, thus for each i a set of strong influences is defined, S_i. With the concept of influence/dependence defined, next two heuristic criteria are given to guide coarse grid selection, C.

1. For each point j that strongly influences an F-point i, j is either a C-point or is strongly influenced by a C-point k that also strongly influences point i.
2. The set of coarse points C should be a maximal subset of all points such that no C-point strongly depends on another C-point.

The first criteria is used to ensure the quality of interpolation and is thus enforced, while the second criteria is used as a guide to limit the coarse grid size. Next the interpolation operator I_{k+1}^k is constructed to interpolate the error for F-point i from neighbouring points. These neighbouring points are split into

three groups being: coarse strong influences (S_i^C), fine strong influences (S_i^F) and points that do not strongly influence i (W_i). The interpolation is given by,

$$\left(I_{k+1}^k \mathbf{e}\right)_i = \begin{cases} e_i & \text{if } i \in C \\ \sum\limits_{j \in S_i^C} \omega_{ij} e_j & \text{if } i \in F \end{cases} \tag{14}$$

where e_i is the error at point i and ω_{ij} is the interpolation weight that determines the contribution from neighbouring points given as

$$\omega_{ij} = -\frac{a_{ij} + \sum\limits_{m \in S_i^F} \left(\frac{a_{im} a_{mj}}{\sum\limits_{k \in S_i^C} a_{mk}} \right)}{a_{ii} + \sum\limits_{n \in W_i} a_{in}} \tag{15}$$

Following the construction of coarse levels is the *solve phase*, where a V-cycle is recursively performed as follows:

Multigrid V-cycle(A^k,b^k,x^k)

1. If $k \neq M$ then:
   (a) Smooth $A^k x^k = b^k$ with μ_1 iterations of a Gauss-Seidel Solver.
   (b) Set $r^{k+1} = I_k^{k+1}\left(b^k - A^k x^k\right)$.
   (c) Apply V-cycle on level $k+1$, V-cycle(A^{k+1},r^{k+1},e^{k+1}).
   (d) Correct the solution, $x^k = x^k + I_{k+1}^k e^{k+1}$.
   (e) Smooth $A^k x^k = b^k$ with μ_2 iterations of a Gauss-Seidel Solver.
2. Otherwise, smooth $A^M x^M = b^M$ with μ_3 iterations of a Gauss-Seidel Solver.

The first challenge associated with the development of the AMG object arose from the requirement that all algorithms were to be of optimal $O(N)$ complexity. Of specific interest are the coarsening and matrix multiplication algorithms. The former was effected by taking advantage of an efficient data storage scheme in which the minimum number of nodes are accessed during each coarsening step. Matrix multiplication, used when applying the Galerkin approach to obtaining coarse grid operators [25], required special attention due to the compressed

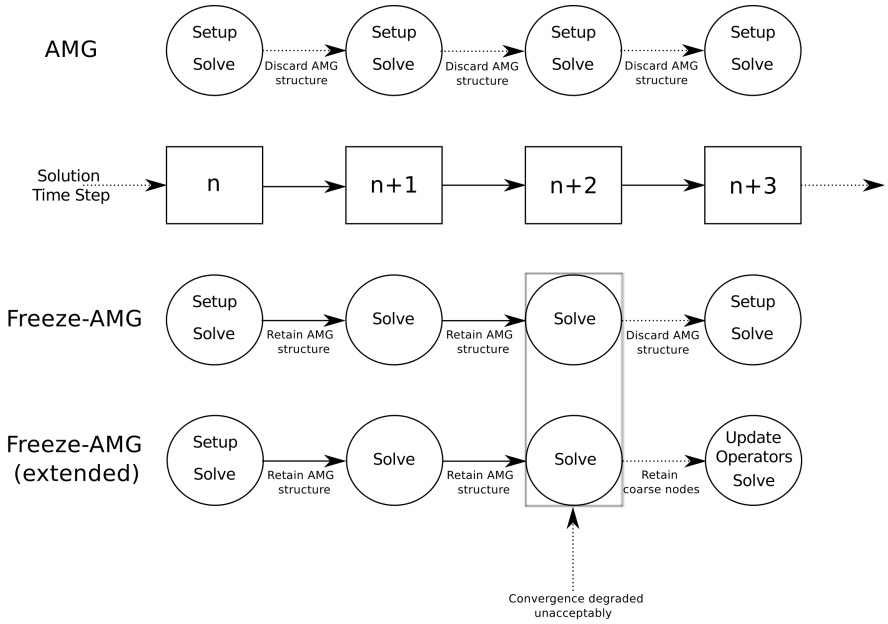

Figure 1: Comparison of Freeze-AMG and C-AMG methodologies

row format employed. A modified matrix multiplication algorithm, which accesses coefficients row-by-row, and eliminates expensive column searches was implemented to address the latter. In order to ensure that linear scaling was achieved, the solver was evaluated on a single time step of the two-dimensional dam-break problem in Section 4.1. Employing structured meshes ranging three orders of magnitude in size, the desired linear complexity was achieved (as depicted in Figure 2).

3.2. Freeze-AMG

A major contributor to time taken by the C-AMG method is the construction of the AMG solver structure, which is comparable to the time spent during solution cycles. While the method is applicable to FSM problems as-is, the fact that the coefficient matrix in Equation (12) changes as a function of time necessitates continuous reconstruction of the AMG solution structure. In the case of free-surface flows, the aforementioned changes in the coefficient matrix are however often localised, since these changes are due to the evolving fluid

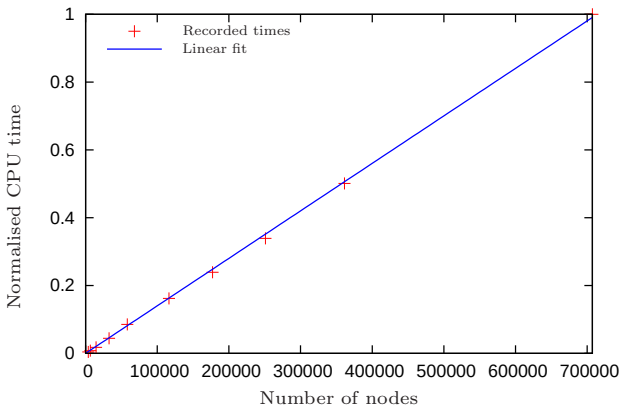

Figure 2: Solver scaling

interface, which is itself a local phenomenon. A given AMG solution structure can therefore be expected to converge for coefficient matrices over several time steps, provided the matrix does not depart too far from the original [26, 27]. As time progresses however, the coefficient matrix would alter to the point where the latter is no longer the case.

The above observations gave rise to the concept of only performing periodic AMG setup steps, thus 'freezing' a given AMG solution structure until convergence is no longer satisfactory. The basic heuristic used in this work to measure this was simply the convergence rate between solver iterations. In addition, a further refinement of the Freeze method was made. This was due to the fact that the coarsening procedure during AMG setup consumed a large part of computational effort. Therefore, the concept of performing only a *partial* setup when convergence degrades was implemented. Specifically, a previously obtained coarsening would be used while only updating coefficient magnitudes. These AMG refinements were dubbed *Freeze-AMG* (F-AMG) and Extended F-AMG, and are illustrated in Figure 1.

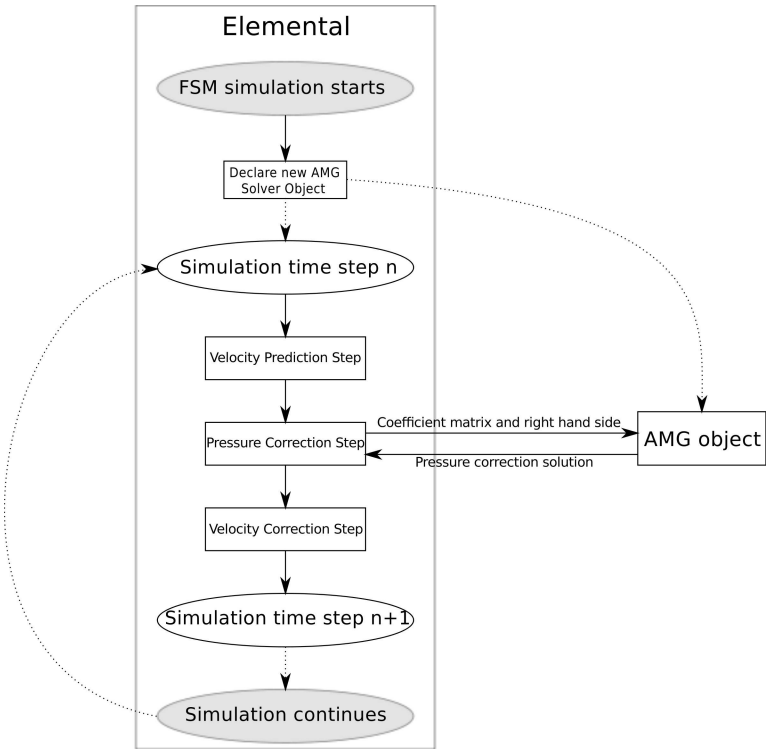

Figure 3: Schematic of interaction between the solver and Elemental

4. SOLVER EVALUATION

In order to evaluate the performance of the solver in terms of efficiency and computational speed improvements, it was applied to a two- and three-dimensional dam-break benchmark problem [28]. This case was selected as it represents a typical fluid sloshing problem with a rapid changing pressure coefficient matrix (due to the high speed of the fluid interface). This involves a greater need for AMG setups and allows more rigorous testing of the robustness and speed of the proposed freeze methodology. In the interests of a rigorous evaluation the problem was solved on a wide range of mesh sizes (two- and three-dimensional) and the CPU cost compared to that achieved if using a competing matrix-free sparse solver. As such, C-AMG, F-AMG and Extended F-AMG were pitted against the preconditioned versions of the Conjugate Gradient (CG), Bi-Conjugate Gradient Stabilised (BiCGSTAB) and GMRES solvers. Various preconditioners were considered, and Lower-Upper Symmetric-Gauss-Seidel (LU-SGS) found to be a matrix-free algorithm which offers speed and efficiency [6]. In addition, solution time scaling as a function of mesh size of the solvers was evaluated and compared. The following was adhered to for all simulations:

- The pressure equation was considered solved only once the scaled residual had been reduced by five orders of magnitude,
- In the interest of automation, the AMG solver was allowed to construct coarse levels until the coarsest level contained no less than 3 rows, with a coarsening threshold (θ [25]) of 0.5,
- A V-cycle was employed for solution, which consisted of $\mu_1 = 2$, $\mu_2 = 2$ and $\mu_3 = 3$ smoothing iterations,
- The convergence rate of V-cycles allowed before a solution structure was deemed inappropriate (i.e. re-coarsening was required) was set as:

$$\frac{\|\mathbf{b} - \mathbf{A}\Delta\mathbf{p}\|_2^i}{\|\mathbf{b} - \mathbf{A}\Delta\mathbf{p}\|_2^{i+1}} < 0.8,$$

where $\| \bullet \|_2$ is the l^2–Norm and i is the iteration number.

The analyses were done on a Dell XPS L702x computer, with a 2^{nd} generation Intel Core i7-2720 CPU and 8Gb of 1,333Mhz RAM.

4.1. Two-Dimensional Dam-break

The solver performance was evaluated by application to the canonical two-dimensional dam-break [28]. Here, a column of water of width $a = 0.146m$ and height $2a$ is allowed to flow under the influence of gravity, as shown in Figure 4. All four boundaries are treated as no-slip. This problem exhibits violent interface motion, as seen in the snapshots of the evolving interface. As such it offers the opportunity to test the robustness of the developed AMG setup methodologies. This is of special interest since the matrix coefficients in Equation (12) will undergo large changes between time steps as the interface sweeps over the domain. Note that for the purpose of this work, a 0.5 second simulation interval was modelled as it contains the time period of the problem characterised by the fastest moving interface. Subsequent to this, the interface settles down to a large degree.

As noted previously, robustness was assessed by checking that the solver reduced the l^2–Norm residual by 5 orders of magnitude. This was indeed the case throughout. Validation consisted of comparing computed to experimental [28] and published results. These are shown schematically in Figure 5 where the results obtained from a 3,710 node structured mesh (Figure 6) are shown to agree well with the available data. Refined mesh sizes converged to the same solution. Following this, the solver comparisons were performed on two-dimensional structured and unstructured meshes ranging from 14,795 to 59,853 nodes. The results obtained are shown in Figure 7. The AMG methods clearly outperform the other solvers, with marginal differences between the F-AMG methods. Further, where F-AMG shows fairly consistent performance over the entire time period, GMRES suffers from higher CPU cost at 0.3s, which coincides with the interface reaching the opposite wall, as shown in Figure 4. Here, interface

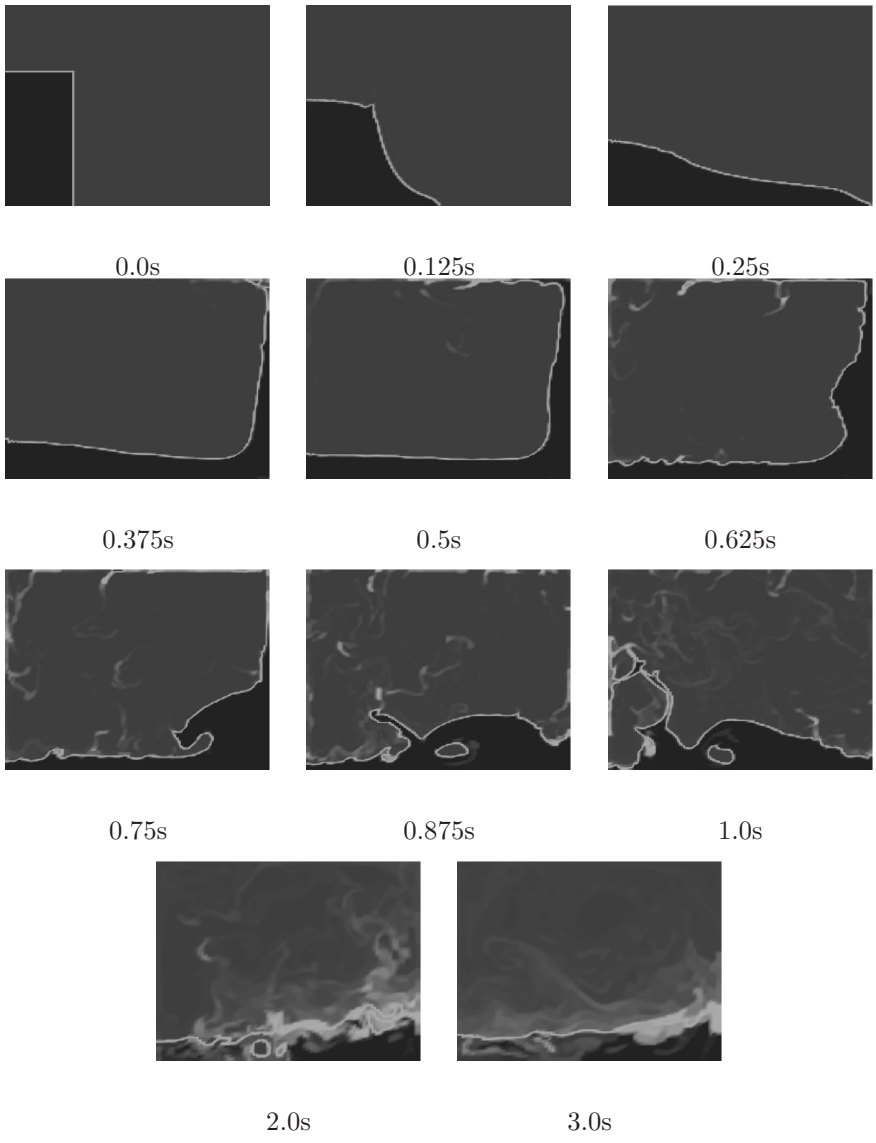

Figure 4: Two-dimensional dam-break - Interface evolution

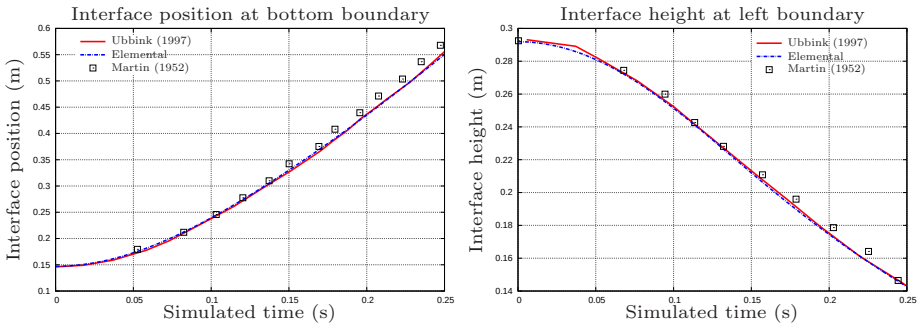

Figure 5: Two-dimensional dam-break - Solution verification

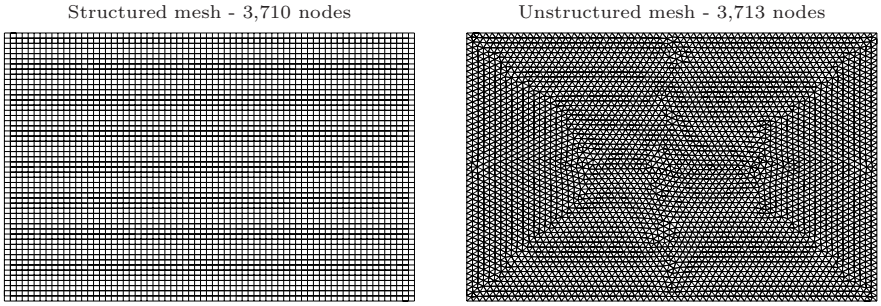

Figure 6: Two-dimensional dam-break structured and unstructured mesh

motion starts to become much more rapid. F-AMG again offers the best performance, noticeably improving on C-AMG. It is also interesting to note how CG improves in relative performance on the unstructured mesh (as compared to structured). This is due to the increase in complexity which favours simpler methods.

Also depicted in Figure 7 is solver scaling as a function of problem size. The data again shows the AMG methods exhibiting superior performance, approaching optimal $O(N)$ complexity. In contrast the other solvers achieved more non-linear scaling.

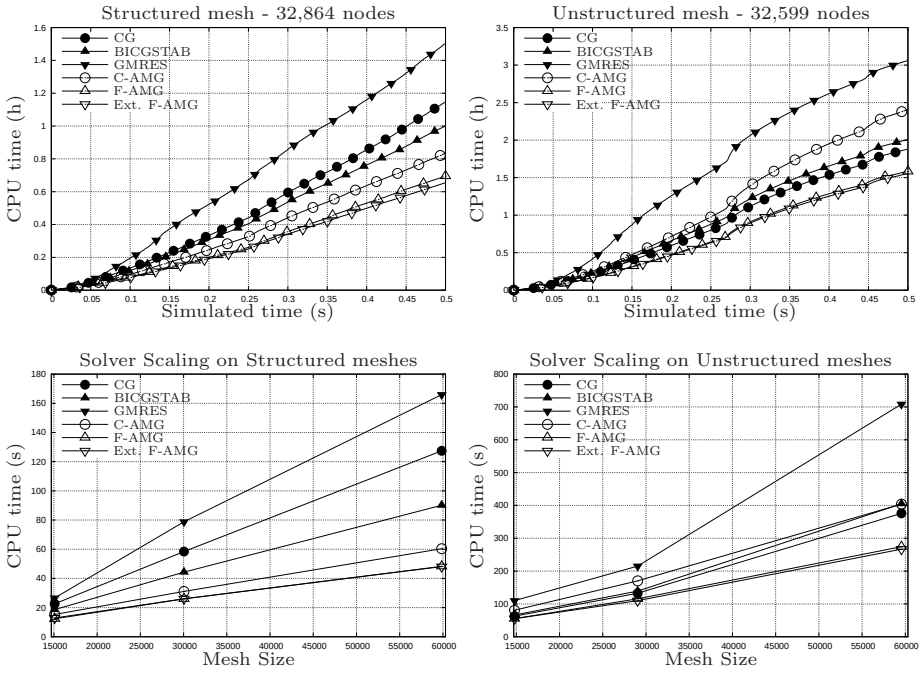

Figure 7: Two-dimensional dam-break solver comparison and scaling

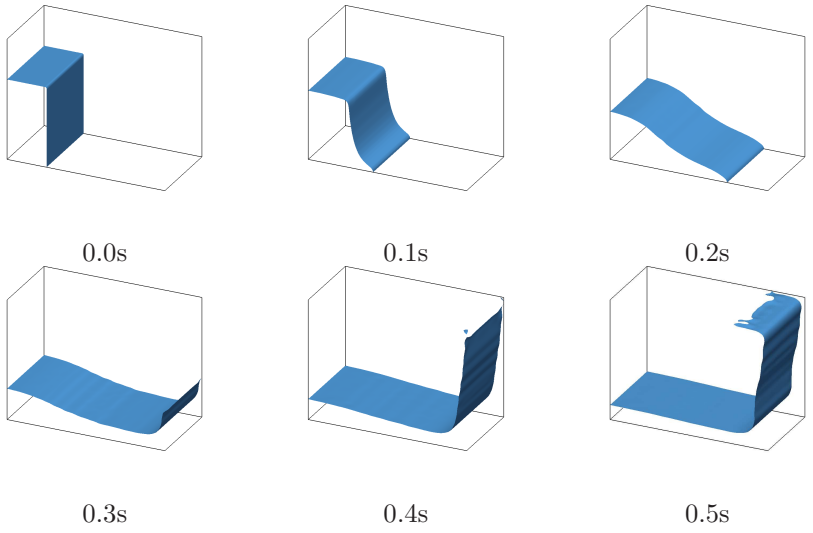

Figure 8: Three-dimensional dam-break interface evolution

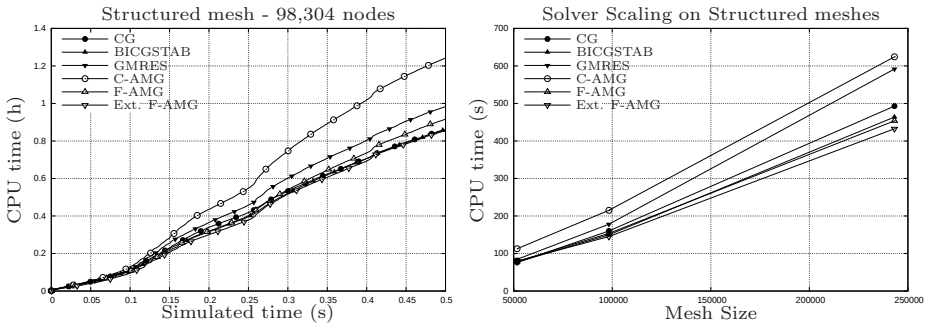

Figure 9: Three-dimensional dam-break solver comparison and scaling

4.2. Three-Dimensional Dam-break

Is it important to evaluate the AMG methodologies on a three-dimensional problem, as the additional dimension adds computational complexity due to increased node-node connectivity. The AMG coarsening algorithm thus becomes more costly and provides further reason to minimise the number of AMG setup steps. The three-dimensional mesh was created by extruding the two-dimensional case into the third dimension. As before, a column of water of width $a = 0.146m$, height and depth $2a$ is allowed to flow under the influence of gravity, as shown in Figure 8. Again, all boundaries are treated as no-slip and 0.5 seconds is simulated.

The solvers were compared on three structured meshes ranging from 51,714 to 243,165 vertexes (isotropic hex elements were employed). The results are depicted in Figure 9. Similar to previously, F-AMG performs favourably as compared to the other solvers. The cost of AMG setups due to increased complexity is evident as C-AMG performs worst. Extended F-AMG in contrast remains competitive and compares well to preconditioned CG and preconditioned BiCGSTAB.

5. CONCLUSIONS

This paper dealt with reducing the CPU cost associated with the solution of the pressure correction equation in a free surface modelling context. For this

purpose, a classic AMG solver was successfully implemented as a plug-in to the *Elemental*$^{\text{TM}}$package. A modular approach was employed, with special care devoted to ensuring optimal algorithm scaling. This solver was extended using two Freeze methodologies in order to mitigate the AMG setup overhead associated with changing coefficient matrices arising in FSM problems. These automatically ensure that AMG setup occurs periodically and only as needed. In order to evaluate the developed solver, it was applied to a two- and three-dimensional dam-break benchmark problem. Simulation cost was then compared to that achieved if using other competing preconditioned matrix-free solvers *viz.* Conjugate Gradient (CG), BiConjugate Gradient Stabilised (BiCGSTAB) and GMRES. The study revealed that the developed Freeze-AMG methods consistently achieved either superior or competitive performance on structured and unstructured meshes. The Freeze-AMG methods exhibited optimal linear scaling with problem size while achieving significant improvements in speed as compared to the classic AMG method. The latter was particularly pronounced on three-dimensional meshes.

References

[1] C. W. Hirt, B. D. Nichols, Volume of fluid (VoF) method for the dynamics of free boundaries, Journal of Computational Physics 39 (1) (1981) 201–225.

[2] S. Osher, J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations, Journal of Computational Physics 79 (1) (1988) 12–49.

[3] O. Ubbink, R. I. Issa, A method for capturing sharp fluid interfaces on arbitrary meshes, Journal of Computational Physics 153 (1) (1999) 26–50.

[4] P. Nithiarasu, An efficient artificial compressibility (AC) scheme based on the characteristic based split (CBS) method for incompressible flow, In-

ternational Journal for Numerical Methods in Engineering 56 (13) (2003) 1815–1845.

[5] O. F. Oxtoby, A. G. Malan, A matrix-free, implicit, incompressible fractional-step algorithm for fluidstructure interaction applications, Journal of Computational Physics 231 (16) (2012) 5389–5405.

[6] A. G. Malan, J. P. Meyer, R. W. Lewis, Modelling non-linear heat conduction via a fast matrix-free implicit unstructured-hybrid algorithm, Computer Methods in Applied Mechanics and Engineering 196 (45) (2007) 4495–4504.

[7] J. A. Heyns, A. G. Malan, T. M. Harms, O. F. Oxtoby, A weakly compressible free-surface flow solver for liquid-gas systems using the volume-of-fluid approach, Journal of Computational Physics 240 (2013) 145–157.

[8] O. F. Oxtoby, A. G. Malan, J. A. Heyns, A computationally efficient 3D finite-volume scheme for violent liquid-gas sloshing, International Journal for Numerical Methods in Fluids 79 (6) (2015) 306–321.

[9] S. Kurioka, D. R. Dowling, Numerical simulation of free surface flows with the level set method using an extremely high-order accuracy WENO advection scheme, International Journal of Computational Fluid Dynamics 23 (3) (2009) 233–243.

[10] J. W. Lee, M. D. Teubner, J. B. Nixon, P. M. Gill, Development of a 3D non-hydrostatic pressure model for free surface flows, in: R. May, A. J. Roberts (Eds.), Proc. of 12th Computational Techniques and Applications Conference CTAC-2004, Vol. 46, 2005, pp. C623–C636.

[11] H. A. Van der Vorst, Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM Journal on scientific and Statistical Computing 13 (2) (1992) 631–644.

[12] M. B. Koçyigit, R. A. Falconer, B. Lin, Three-dimensional numerical modelling of free surface flows with non-hydrostatic pressure, International Journal for Numberical Methods in Fluids 40 (9) (2002) 1145–1162.

[13] O. Soto, R. Löhner, F. Camelli, A linelet preconditioner for incompressible flow solvers, International Journal of Numerical Methods for Heat & Fluid Flow 13 (1) (2003) 133–147.

[14] R. Aubry, F. Mut, R. Löhner, J. R. Cebral, Deflated preconditioned conjugate gradient solvers for the Pressure-Poisson equation, Journal of Computational Physics 227 (24) (2008) 10196–10208.

[15] L. Štrubelj, I. Tiselj, B. Mavko, Simulations of free surface flows with implementation of surface tension and interface sharpening in the two-fluid model, International Journal of Heat and Fluid Flow 30 (4) (2009) 741–750.

[16] J. Farmer, L. Martinelli, A. Jameson, Fast multigrid method for solving incompressible hydrodynamic problems with free surfaces, AIAA journal 32 (6) (1994) 1175–1182.

[17] J. Waltz, R. Löhner, A grid coarsening algorithm for unstructured multigrid applications, in: 38th AIAA Aerospace Sciences Meeting and Exhibit, 2000.

[18] S. Popinet, Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries, Journal of Computational Physics 190 (2) (2003) 572–600.

[19] R. Löhner, Projective prediction of pressure increments, Communications in Numerical Methods in Engineering 21 (4) (2005) 201–207.

[20] D. Kim, P. Moin, Pressure-correction algorithm to solve Poisson systems with constant coefficients for fast two-phase simulations, Center for Turbulence Research Annual Research Briefs.

[21] J. W. Ruge, K. Stüben, Algebraic multigrid, in: S. McCormick (Ed.), Multigrid Methods, Vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, 1987, pp. 73–130.

[22] A. G. Malan, R. W. Lewis, On the development of high-performance C++ object-oriented code with application to an explicit edge-based fluid dynamics scheme, Computers & Fluids 33 (10) (2004) 1291–1304.

[23] O. C. Zienkiewicz, R. L. Taylor, The Finite Element Method: Volume 3 - Fluid Dynamics, 5th Edition, Butterworth-Heinemann, Oxford, 2000.

[24] A. G. Malan, R. W. Lewis, An artificial compressibility CBS method for modelling heat transfer and fluid flow in heterogeneous porous materials, International Journal for Numerical Methods in Engineering 87 (1-5) (2011) 412–423.

[25] W. L. Briggs, V. E. Henson, S. F. McCormick, A Multigrid Tutorial, Second Edition, SIAM, Society for Industrial and Applied Mathematics, 3600, University City Science Center, Philadelphia, PA, 2000.

[26] K. Stüben, Algebraic multigrid (AMG): an introduction with applications, GMD-Forschungszentrum Informationstechnik, 1999.

[27] C. Sun, Parallel algebraic multigrid for the Pressure Poisson equation in a finite element Navier-Stokes solver, Ph.D. thesis, Rensselaer Polytechnic Institute (2008).

[28] J. C. Martin, W. J. Moyce, Part IV. an experimental study of the collapse of a liquid column on a rigid horizontal plane, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 244 (882) (1952) 312–324.