Application of multifrontal and GMRES solvers for multi-size particulate flow in rotating channels

Pankaj K. Gupta

Department of Mechanical Engineering,
MVGR College Chintalavalasa,
Vizianagaram, 535005, India
E-mail: pankajkgupta@gmail.com

Krishnan V. Pagalthivarthi*

GIW Industries, Inc.,
Grovetown, GA 30813, USA
Fax: +1-706-868-8025
E-mail: krish18pag@yahoo.com
*Corresponding author

Abstract: The study deals with the quantification of computational demands of predicting volume-averaged multi-size particulate flow in rotating 2D channels using Galerkin finite element method. The performances of multifrontal (UMFPACK 4.4) direct solver and preconditioned recursive Generalised Minimum Residual (GMRES) solver are compared. For problems with large number of elements, GMRES is found to take 70–80% less computation time than the direct multifrontal solver. With a view to studying 3D problems, comparisons between preconditioned ILU(k) and GMRES are also included for non-linear heat conduction. Results show that GMRES could be competitive for 3D problems as well under certain circumstances.

Keywords: GMRES; preconditioner; UMFPACK 4.4; multifrontal; multi-size particulate flow; rotating channel.

Reference to this paper should be made as follows: Gupta, P.K. and Pagalthivarthi, K.V. (2007) 'Application of multifrontal and GMRES solvers for multi-size particulate flow in rotating channels', *Progress in Computational Fluid Dynamics*, Vol. 7, No. 6, pp.323–336.

Biographical notes: Pankaj K. Gupta earned his Bachelor's Degree in Mechanical Engineering from the University BDT College of Engineering in 1996. He completed his Master of Science (Research), MS (R) in Applied Mechanics from Indian Institute of Technology, Delhi in 2001. He has recently submitted his Doctoral Thesis to the Department of Applied Mechanics at IIT Delhi. His research interests include two-phase flow and computational fluid dynamics.

Krishnan V. Pagalthivarthi is currently CFD research leader at the Hydraulics Laboratory of GIW Industries, Inc., Grovetown, Georgia, USA. Formerly, he was Professor of Applied Mechanics at the Indian Institute of Technology, Delhi. He received his Bachelor's Degree in 1979 from Indian Institute of Technology, Delhi and his MS and PhD Degrees in 1984 and 1988, respectively, from Georgia Institute of Technology, USA – all in Mechanical Engineering. His research interests include two-phase flow, FE-CFD and erosion. He is recognised as a CFD software developer for two-phase flow and erosion prediction and for his excellence in teaching.

1 Introduction

Numerical simulation of Navier-Stokes equations involves a large set of non-linear equations, requiring high computing power in terms of speed and memory. In multiphase particulate flow problems, the computing demands are further intensified. Particulate flows are usually modelled using one of three different techniques – Eulerian-Eulerian (E-E) (Slater and Young, 2001; Slater et al., 2003), Eulerian-Lagrangian (E-L) (Sommerfeld and Huber, 1999; Patankar and Joseph, 2001) and Direct Numerical Simulation (DNS) (Glowinski et al., 1999). The E-E approach (used in this study) treats the particulate phase and the continuous phase as interpenetrating continua exchanging mass, momentum and energy with each other. Conservation equations (derived by spatial, time or ensemble averaging) are solved separately for the carrier fluid and for each particulate phase.

Multi-size particulate flow (with six size classes) through a two-dimensional rectangular channel has been studied recently (Gupta and Pagalthivarthi, 2006;

Pagalthivarthi et al., 2005) using Galerkin finite element method. In this case, a total of 21 field unknowns (two velocity components for each particle species and for the mixture, one concentration variable for each particle species, and the pressure that is assumed common to all phases) are involved. The resulting set of weak form (algebraic) equations may be solved using either a direct solver or an iterative solver. The direct solution methods generally involve the use of frontal algorithms (Irons, 1970) in finite element applications. The multifrontal method, found attractive for large spare systems, is a generalisation of the frontal method and was originally developed for symmetric systems (Amestoy and Duff, 1989). Subsequently, an unsymmetric multifrontal algorithm (UMFPACK) was developed (Davis and Duff, 1997) for general sparse unsymmetric matrix. The advent of multifrontal solvers has greatly increased the efficiency of direct solvers for sparse systems. They make full use of the high performance computer architecture by invoking the level three Basic Linear Algebra Subprograms (BLAS) library. Thus memory requirement is greatly reduced and the computing speed is greatly enhanced.

Solution of multi-size particulate flow through a horizontal channel was reported in an earlier paper (Pagalthivarthi et al., 2005) using unifrontal solver. However, owing to high memory requirements and relatively low computational speed of the unifrontal algorithm, very fine meshes could not be used in that study with the available computing resources (2 GB RAM, 2 GHz Pentium 4 machines). In a subsequent study (Gupta and Pagalthivarthi, 2006) involving multi-size particulate flow through rotating channels, UMFPACK 4.4 multi-frontal solver was employed with significant improvements in memory requirement and computational speed, allowing much finer meshes to be used.

All direct solvers still involve huge computational costs during the factorisation step, especially when a large number of nodal degrees of freedom (or elements) are involved. A further improvement in the computational speed may be possible using iterative solvers, allowing very large problems to be attempted. In fact, both direct and iterative solvers can be effectively combined (Einset and Jensen, 1992) to generate computationally efficient codes. Following this idea, the present study uses a recursive GMRES iterative solver in combination with a multifrontal direct solver (UMFPACK 4.4) as the preconditioner. While solving non-linear problems, preconditioners can be efficiently developed for direct solvers having the capability of storing the LU factors for further reuse. In this study, the LU factors obtained from the direct solver during the first iteration are reused as a preconditioner for the subsequent iterations of the GMRES solver.

A variety of preconditioning methods are available for enhancing the convergence of iterative solvers such as GMRES, BiCGSTAB and ILU (Saad, 2003). For instance, the ILU preconditioner and its variants like MILU and ILUT work very well for well-conditioned matrices. But in the case of finite element solution of fluid flow, the Jacobian matrix is often poorly conditioned (Adams, 1999). Specifically, the Galerkin *u-v-p* (combined primitive variable) formulation of the coupled incompressible Navier-Stokes equations yields rather ill-conditioned equations. The element matrix continuity equations contain zeros in the positions corresponding to the pressure terms. To overcome this, a segregated approach based on pressure projection techniques (Reddy and Gartling, 2001) is used in which the ILU solver is applied to the pressure and the momentum equations separately. Although on the one hand this greatly reduces the memory requirement (because of ILU), on the other it loses the advantage of quadratic convergence offered by the fully coupled Newton's method (resulting from a coupled *u-v-p* formulation). Moreover, literature suggests (Kurreck and Wittig, 1997) that fully coupled formulations are more robust and stable. Therefore, preconditioners such as ILU have not been used for the fully coupled formulation used in this study.

The multigrid method pioneered by Brandt (1997) is a powerful technique for the iterative solution of discretised elliptic partial differential equations. Multigrid techniques have been used for solving Navier-Stokes problems using stream function and vorticity equations (Ghia et al., 1982) and the primitive variables (Phillips and Schmidt, 1985). The coupling of pressures and velocities is observed to speed up convergence (Vanka, 1985). A multigrid technique using a Symmetrical Coupled Gauss-Seidel (SCGS) technique as a relaxation procedure has been reported (Vanka, 1986) for calculating three-dimensional recirculating flows. The CPU times are reported to vary linearly with the increase in grid fineness as is characteristic of multigrid techniques.

Given the complexity of implementation of ILU and multigrid techniques for Navier-Stokes problems and the easy accessibility of multifrontal codes for any general sparse system of linear equations of the form $AX = B$, it is tempting to develop GMRES-type iterative solvers with preconditioner generated from the multifrontal direct solver. This strategy is employed in this study for handling the computational load of multisize particulate flow (E-E formulation). It should be noted that multigrid techniques can also be developed (not done in this study) in conjunction with the multifrontal preconditioning being applied at the coarser mesh levels. This could greatly enhance the computational efficiency in terms of both speed and memory.

The main subject of this paper is to compare the performance of a preconditioned GMRES solver with that of the multifrontal direct solver, particularly in the context of multi-size particulate flows. First, the simple case of developing laminar flow in a channel is studied, and for reference, the results are compared with that of a unifrontal solver for different mesh sizes. The multifrontal direct and GMRES solvers are then applied to multi-size particulate flow in a rotating channel.

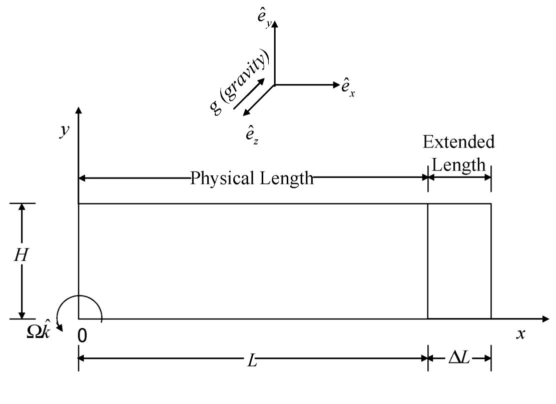
For the multiphase flow in a rotating channel, a sequential approach is used to solve the system of equations (resulting from GFEM formulation) governing the mixture flow and individual particulate species momentum and mass continuity. Using the primitive variable (u_m, v_m, p_m) coupled formulation, the mixture flow field is computed first, followed by the two momentum equations of each particle species. This is followed by the computation of the concentration field of each species governed by a separate convection-diffusion equation. Computations of various fields are repeated until all the field variables (of mixture flow, solid species and concentration) converge to a prescribed tolerance of 10^{-6} in the infinity norm of the correction vector.

The multifrontal direct solver (either used independently or used as the preconditioner of GMRES) needs huge memory, especially for very fine meshes and for 3D computations. To address this issue, the memory requirement of multifrontal solvers for 2D Navier-Stokes problems has been examined in this study. Finally, the feasibility of using multifrontal solver for 3D problems is also analysed through the example of 3D non-linear heat conduction problem.

2 Mathematical formulation

The governing equations for momentum and mass conservation for the mixture and the individual species of solids in a rotating reference frame are discussed in Gupta and Pagalthivarthi (2006). The schematic of a two-dimensional channel rotating in an orthogonal mode at constant angular velocity $\Omega\hat{k}$ is shown in Figure 1.

Figure 1 Schematic of the rotating channel domain



The continuity and momentum equations for the mixture are written in a non-dimensional scalar form as

$$\frac{\partial}{\partial x}(S_m u_m)+\frac{\partial}{\partial y}(S_m v_m)=0, \tag{1}$$

$$\begin{aligned}\frac{\partial}{\partial x}(S_m u_m u_m)+\frac{\partial}{\partial y}(S_m u_m v_m)&=-\frac{\partial p}{\partial x}+S_m(\Omega^2 x+2\Omega v_m)\\&+\frac{\partial}{\partial x}\left(\frac{\mu_m^*+\mu_{mt}^*}{\mathrm{Re}}\right)\left\{\frac{4}{3}\frac{\partial u_m}{\partial x}-\frac{2}{3}\frac{\partial v_m}{\partial y}\right\}\\&+\frac{\partial}{\partial y}\left(\frac{\mu_m^*+\mu_{mt}^*}{\mathrm{Re}}\right)\left\{\frac{\partial u_m}{\partial y}+\frac{\partial v_m}{\partial x}\right\},\end{aligned} \tag{2}$$

and

$$\begin{aligned}\frac{\partial}{\partial x}(S_m u_m v_m)+\frac{\partial}{\partial y}(S_m v_m v_m)&=-\frac{\partial p}{\partial y}+S_m(\Omega^2 y-2\Omega u_m)\\&+\frac{\partial}{\partial x}\left(\frac{\mu_m^*+\mu_{mt}^*}{\mathrm{Re}}\right)\left\{\frac{\partial u_m}{\partial y}+\frac{\partial v_m}{\partial x}\right\}\\&+\frac{\partial}{\partial y}\left(\frac{\mu_m^*+\mu_{mt}^*}{\mathrm{Re}}\right)\left\{\frac{4}{3}\frac{\partial v_m}{\partial y}-\frac{2}{3}\frac{\partial u_m}{\partial x}\right\},\end{aligned} \tag{3}$$

where u_m, v_m are the x and y components of mixture velocity, p is the pressure, Ω is the rotation rate, $\mu_m^*=\mu_m/\mu_L$, $\mu_{mt}^*=\mu_{mt}/\mu_L$, μ_m and μ_L are the laminar viscosities of the mixture and carrier liquid respectively and μ_{mt} is the eddy viscosity of the mixture. The bulk flow Reynolds number, Re, is computed as $\mathrm{Re}=(\rho_L U_0 L)/\mu_L$, with U_0 being the inlet velocity, ρ_L the carrier density, L the channel length, $S_m=\rho_m/\rho_L$, and ρ_m being the mixture density. Velocities are non-dimensionalised with respect to U_0, pressure with respect to $\rho_L U_o^2$, rotation rate with respect to U_0/L, lengths with respect to L and densities with respect to ρ_L. The eddy viscosity is computed based on a $k-\varepsilon$ model for the carrier, modified for rotation and concentration.

The x and y momentum equations for the particulate size class 'k' are

$$\begin{aligned}\frac{\partial}{\partial x}(C_k u_k u_k)+\frac{\partial}{\partial y}(C_k v_k u_k)+\frac{C_k}{S_S}\frac{\partial p}{\partial x}&=C_k(\Omega^2 x+2\Omega v_k)\\&+\frac{3C_D}{4d_{pk}K_{Sk}S_S}\left|\vec{u}_m-\vec{u}_k\right|(u_m-u_k)C_k(1-C)^{-1.65}\\&+\frac{1}{\mathrm{Re}}\left[\frac{\partial}{\partial x}C_k\left\{\begin{matrix}\frac{\upsilon_{kE}}{\upsilon_L}\left(\frac{4}{3}\frac{\partial u_k}{\partial x}-\frac{2}{3}\frac{\partial v_k}{\partial y}\right)\\-\frac{\upsilon_{kt}}{\upsilon_L\tan\theta}\left(\frac{\partial u_k}{\partial y}+\frac{\partial v_k}{\partial x}\right)\end{matrix}\right\}\right]\\&+\frac{1}{\mathrm{Re}}\left[\frac{\partial}{\partial y}C_k\frac{\upsilon_{kE}}{\upsilon_L}\left(\frac{\partial u_k}{\partial y}+\frac{\partial v_k}{\partial x}\right)\right],\end{aligned} \tag{4}$$

and

$$\begin{aligned}\frac{\partial}{\partial x}(C_k u_k v_k)+\frac{\partial}{\partial y}(C_k v_k v_k)+\frac{C_k}{S_S}\frac{\partial p}{\partial y}&=C_k(\Omega^2 y-2\Omega u_k)\\&+\frac{3C_D}{4d_{pk}K_{Sk}S_S}\left|\vec{u}_m-\vec{u}_k\right|(v_m-v_k)C_k(1-C)^{-1.65}\\&+\frac{1}{\mathrm{Re}}\left[\frac{\partial}{\partial x}C_k\frac{\upsilon_{kE}}{\upsilon_L}\left(\frac{\partial u_k}{\partial y}+\frac{\partial v_k}{\partial x}\right)\right]\\&+\frac{1}{\mathrm{Re}}\left[\frac{\partial}{\partial y}C_k\left\{\begin{matrix}\frac{\upsilon_{kE}}{\upsilon_L}\left(\frac{4}{3}\frac{\partial v_k}{\partial y}-\frac{2}{3}\frac{\partial u_k}{\partial x}\right)\\-\frac{\upsilon_{kt}}{\upsilon_L\tan\theta}\left(\frac{\partial u_k}{\partial y}+\frac{\partial v_k}{\partial x}\right)\end{matrix}\right\}\right],\end{aligned} \tag{5}$$

where u_k, v_k are the x and y components of solid velocity, C_k is the volumetric concentration of species 'k', S_S is the solids specific gravity (assumed equal for all size classes), d_{pk} is the particle diameter of *k*th class, K_{Sk} is the particle shape factor, υ_{kE} and υ_{kt} are, respectively, the effective kinematic viscosity and kinematic eddy viscosity of the *k*th size class, tan θ is the dynamic friction coefficient (to account for particle collisional stresses (Gupta and Pagalthivarthi, 2006; Pagalthivarthi et al., 2005) and υ_L is the kinematic viscosity of the liquid.

The concentration distribution of each size class is governed by a convection-diffusion equation,

$$\frac{\partial}{\partial x}(C_k u_k) + \frac{\partial}{\partial y}(C_k v_k) = \frac{\partial}{\partial x}\varepsilon_{kx}\frac{\partial C_k}{\partial x} + \frac{\partial}{\partial y}\varepsilon_{ky}\frac{\partial C_k}{\partial y}, \qquad (6)$$

where, ε_{kx} and ε_{ky} are the particle eddy diffusivities in the *x* and *y* directions. The equations for these are presented elsewhere (Gupta and Pagalthivarthi, 2006).

The boundary conditions for the mixture and solid species '*k*' are prescribed as follows:

- Along the channel inlet ($x = 0$ and $0 \le y \le H$):

$$u_m = 1;\ v_m = 0;\ u_k = 1;\ v_k = 0. \qquad (7)$$

- Along the channel exit ($x = 1 + \Delta L$ and $0 \le y \le H$):

$$p = 0;\ \frac{\partial u_m}{\partial x} = 0;\ \frac{\partial v_m}{\partial x} = 0;\ \frac{\partial u_k}{\partial x} = 0;\ \frac{\partial v_k}{\partial x} = 0. \qquad (8)$$

- Along the suction and pressure-side walls [$y = 0$ (or $y = H$), $0 \le x \le 1 + \Delta L$], standard wall functions are used such that

$$u^+ = \frac{1}{\kappa}\ln(Ey^+) \text{ when } y^+ > 11.6$$
$$u^+ = y^+ \text{ when } y^+ \le 11.6, \qquad (9)$$

where $u^+ = u/u_\tau$ and $y^+ = y_w u_\tau/\upsilon_L$ (for pure liquid phase). Here, y_w is the distance of the node nearest from the wall, and u_τ is the friction velocity. In computing y^+, υ_L is replaced with υ_m $(= \mu_m/\rho_m)$ for wall functions of mixture flow field, and υ_k $(= \mu_k/\rho_k)$ for the wall functions of solid species '*k*'. In the absence of data for solid–liquid flows, the values of $\kappa = 0.4$ and $E = 9.8$ are considered corresponding to simple fluid flow in smooth pipes (Schlichting, 1979). In an earlier study (Gupta and Pagalthivarthi, 2006), it is demonstrated that the results are not very sensitive to the value of *E*. The concentration field of solid species '*k*' is obtained by solving Equation (6) subject to the boundary conditions:

$$C_k\Big|_{\text{inlet}} = (C_a)_k \text{ and } \frac{\partial C_k}{\partial n}\Bigg|_{\text{exit and walls}} = 0, \qquad (10)$$

where $(C_a)_k$ is the average inlet concentration of species '*k*'.

For laminar flow through a stationary channel, the governing equations of continuity and momentum may be deduced from Equations (1)–(3) by setting $S_m = 1$, $\mu^*_{mt} = 0$, $\mu^*_m = 1$, and $\Omega = 0$.

As stated in the introduction, the crucial concern in using multifrontal direct solvers (as preconditioners in GMRES) is the huge memory requirement. This would be particularly serious for 3D problems. The feasibility of extending the present 2D multisize particulate flow computations to three dimensions may be evaluated with the help of a much simpler problem of non-linear heat conduction in three dimensions. The steady state temperature distribution in a cuboid region is governed by the equation

$$\nabla.(k\nabla T) = 0, \qquad (11)$$

where

$$k = k_0\left(1 - \frac{0.1T}{T_0}\right) \qquad (12)$$

is the temperature (*T*)-dependent thermal conductivity. All faces (except one) of the region are maintained at a specified temperature. The remaining face is maintained at a different specified temperature.

3 Numerical formulation

The numerical algorithm consists of three iterative stages as follows:

1 Compute the mixture velocity field and pressure field using a combined Newton's iteration for u_m, v_m and p (Eqns. 1–3) assuming the velocity field and concentration of each solid species to be fixed for this step.

2 Compute the velocity field of each solid species '*k*' using a combined Newton's iteration for u_k and v_k (Eqns. 4–5) and Equation (3) compute the concentration field C_k of each solid species by Newton's iteration (Eq. 6). In these steps, the eddy viscosity of the mixture, the eddy viscosity for the momentum of the *k*th solid species and the eddy diffusivity for concentration of the *k*th solid species are required. These are computed from the eddy viscosity of the pure liquid flow governed by $k - \varepsilon$ equations with a correction for rotation (Gupta and Pagalthivarthi, 2006). The eddy viscosity field of the pure liquid is then modified for the influence of concentration. The detailed equations for the computation as well as finite element formulation are given in Gupta and Pagalthivarthi (2006) and Pagalthivarthi et al. (2005).

3 Galerkin Finite Element Method (GFEM) is used in all three steps. For step 1, Q_1Q_0 elements are used, the velocity components (u_m, v_m) being interpolated bilinearly and the pressure assumed constant over each element. For steps 2 and 3, bilinear interpolation is used for u_k, v_k and C_k. The successful performance of Q_1Q_0 elements for multi-size particulate flows is detailed in Gupta and Pagalthivarthi (2006) and Pagalthivarthi et al. (2005). Obviously for laminar flow, only step 1 is necessary. For the 3D non-linear heat conduction problem, eight-node hexahedral (trilinear) elements are used.

The non-linear system of equations obtained from GFEM at each step is solved by Newton's method. Let $\underset{\sim}{X}^{(n)}$ be the available vector of field unknowns for the *n*th iteration. Then the update for the (*n* + 1)st iteration is obtained as

$$\underset{\sim}{X}^{(n+1)} = \underset{\sim}{X}^{(n)} + \alpha \delta \underset{\sim}{X}^{(n)}, \tag{13}$$

where α is an under-relaxation factor, and $\delta \underset{\sim}{X}^{(n)}$ is the correction vector obtained by solving the linearised system

$$[J]\{\delta \underset{\sim}{X}^{(n)}\} = -\{\underset{\sim}{R}_X\}^{(n)}. \tag{14}$$

Here, [*J*] is the Jacobian matrix,

$$[J] = \frac{\partial \underset{\sim}{R}_X^{(n)}}{\partial \underset{\sim}{X}^{(n)}}. \tag{15}$$

For the three steps of the iterative computation of mixture velocity, *k*th solid velocity, and *k*th solid concentration, the Galerkin residuals $\underset{\sim}{R}_X$ and field vectors $\underset{\sim}{X}$ are listed in Table 1. Newton's iteration for each step (1, 2 or 3) is continued till the infinity norm of the correction vector $\delta \underset{\sim}{X}^{(n)}$ converges to a prescribed tolerance of 10^{-6}. The non-linear three-dimensional heat conduction problem is solved by applying Picard iteration to the weak (Galerkin) form of the governing equation, with the same convergence criterion as for the flow problems.

Table 1 List of $\underset{\sim}{R}_X$ and $\underset{\sim}{X}$ for mixture, species and concentration

	$\underset{\sim}{R}_X$	$\underset{\sim}{X}$
Mixture velocity	$(\underset{\sim}{R}_u^m, \underset{\sim}{R}_v^m, \underset{\sim}{R}_p^m)^T$	$(\underset{\sim}{u}_m, \underset{\sim}{v}_m, \underset{\sim}{p}_m)^T$
Solid velocity	$\underset{\sim}{R}_u^k, \underset{\sim}{R}_v^k$	$\underset{\sim}{u}_k, \underset{\sim}{v}_k$
Solid concentration	$\underset{\sim}{R}_C^k$	$\underset{\sim}{C}_k$

4 Direct (Frontal) and iterative (GMRES) solvers

The main concern of the present study is the quantitative comparison of the performance of direct (multifrontal) solver and iterative (recursive GMRES) solver for Equation (14).

In the present study, UMFPACK 4.4 (Davis, 2004a, 2004b) is used as the engine for the solution of Equation (14) by multifrontal direct method; it is also used to determine the preconditioner for the GMRES method. UMFPACK consists of a set of FORTRAN callable ANSI/ISO C routines for solving unsymmetric sparse linear systems using Unsymmetric MultiFrontal method. It requires the unsymmetric, sparse matrix (resulting from the finite element formulation) to be input in a sparse (compressed sparse column) format. Different pre-ordering strategies (Davis 2004a, 2004b; Davis et al., 2004a, 2004b) are used to make the solver more memory efficient. The solver depends on BLAS 3 level routines. The performance of UMFPACK 4.4 is significantly better when optimised BLAS routines are used. In the present work, Intel® Math Kernel library is used for optimised BLAS routines.

UMFPACK 4.4 consists of three steps for solving the linear system. In the first step, it performs symbolic factorisation, computing the upper bounds on the non-zeros in *L* and *U*, the floating point operations required, and the memory usage of the factorisation routine. In the second step, numerical factorisation is carried out. In the third step, it solves the linear system using the computed *LU* factors.

For multi-size particulate flow, only the UMFPACK 4.4 multifrontal solver is used in this study. However, for a benchmark comparison for the case of laminar flow of pure liquid, the unifrontal method (Pagalthivarthi and Ramanathan, 2002) is also used.

For obtaining an iterative solution of linear systems, the GMRES with an unsymmetric non-singular matrix is a popular method (Saad and Schultz, 1986). In this study, an efficient and robust variant of GMRES known as GMRESR (Vandervorst and Vuik, 1994) is used. For ready reference, a GMRESR algorithm with preconditioning for solving the linear system of equations $Ax = b$ is presented in the Appendix.

A good preconditioner is essential to accelerate the convergence of iterative methods. A preconditioner is essentially an approximation of the left-hand side matrix whose inverse can be computed more easily than the original matrix itself (Saad, 2003). In the GMRESR algorithm (refer Appendix), an external preconditioner is applied in addition to the internal GMRES preconditioning operation. This is crucial for accelerating its convergence.

In the present work, the *LU* factors obtained from the direct (multifrontal) solver are used to obtain the preconditioner. In the Newton's iterations, the left-hand side matrix is the Jacobian $(\underset{\approx}{J})$, which is updated during each iteration. The first iteration is solved using the UMFPACK 4.4 direct solver and the *LU* factors are stored in the memory. During subsequent iterations, Equation (14) may also be solved using GMRES iterative procedure. The *LU* factors stored for the initial Jacobian $\underset{\approx}{J}^{(0)}$ are used as preconditioner in the subsequent iterations. Since the *LU* factors are already stored, the preconditioning step is relatively inexpensive.

Note that a separate preconditioner is necessary for the mixture velocity field, the *k*th solid velocity field and *k*th solid concentration field, since the Jacobian matrix for each of these computations is different. When the solver goes from one set of equations to the next set of equations, the previously stored *LU* factors are freed from the memory. The new set of *LU* factors is again obtained during the first iteration of the new set of equations. In this way, only one set of *LU* factors is stored in memory at a time.

5 Results and discussion

The case of laminar flow in a channel is considered first to benchmark the performance of multifrontal and GMRES solutions *vis-à-vis* the unifrontal solution.

Subsequently, the multifrontal and iterative solver performances are compared for high-demand computations of multi-size particulate flow. Finally, a comparison of GMRES and ILU is presented for the non-linear 3D heat conduction problem.

5.1 Laminar channel flow

Results are first presented for the standard problem of steady laminar flow through a two-dimensional stationary channel at Reynolds number, Re = 100. The velocity profiles predicted by the direct multifrontal UMFPACK 4.4 solver and GMRES iterative solver are found to be practically identical with the analytic parabolic velocity profiles, with a predicted centreline velocity of $1.5U_0$.

In Figure 2, the convergence history of both the multifrontal direct solver and the iterative GMRES solver are shown for a 100×50 mesh (100 four-noded elements in the x-direction and 50 elements in the y-direction). For the sake of comparison, the performance of unifrontal solver is also included. As noted in the foregoing, for the first iteration, UMFPACK 4.4 is used both for the direct and for the iterative (GMRES) solvers. Hence the solution time for the first iteration using both methods is the same. For each subsequent iteration, the direct solver takes practically the same time as the first iteration. However, there is a significant reduction in the time taken by subsequent iterations of the GMRES solver. This is owing to the fact that the GMRES solver uses the *LU* factors obtained during the first iteration (by direct solver) as a preconditioner. A reduction of 40% computational time is seen in Figure 2 for the overall solution. In comparison to the multifrontal direct solver and the GMRES solver, the unifrontal direct solver is much slower, consuming more than seven times the overall computational time of the multifrontal solver.

Figure 2 Comparison of CPU time and correction updates for different solvers

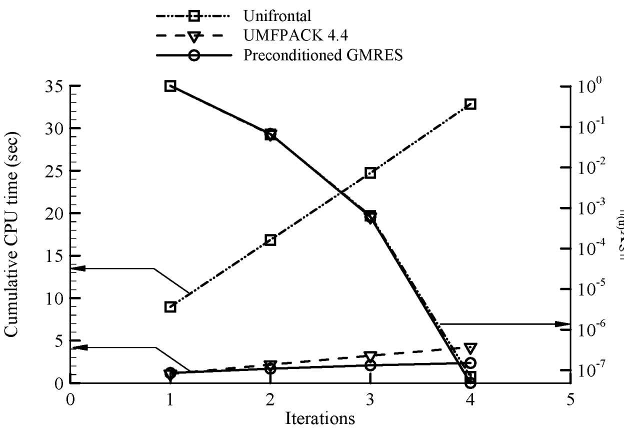


Figure 2 also demonstrates the quadratic convergence rate (characteristic of Newton's method) for all three solvers. Only the time taken per iteration is different from solver to solver. Thus, by using GMRES solver in combination with a preconditioner from the direct solver, the same Newton's convergence rate is obtained in less computational time.

Table 2 shows the performance of the three solvers for different meshes. Both the memory required for storing *LU* factors and the overall (total) computation times for obtaining converged solutions are shown.

Table 2 Comparison of total CPU times (in sec) for the computation of laminar flow using different solvers implementing Newton's formulation

	UMFPACK 4.4 (Direct Multifrontal)		*GMRES*		*Unifrontal*	
Mesh size	*Time* (sec)	*Memory* (MB)	*Time* (sec)	*Memory* (MB)	*Time* (sec)	*Memory* (MB)
100×20	1.12	5.01	0.83	5.01	2.67	12.7
100×50	4.16	24.5	2.55	24.5	33.8	67.8
100×100	10.83	69	6.65	69	300	256.3
200×20	2.29	10	1.81	10	5.64	25.3
200×50	8.55	51.4	5.39	51.4	68	135
200×100	22.11	129	13.56	129	530	510
500×20	5.73	24	4.31	24	16	63
500×50	21.35	119	14.32	119	228	337
500×100	58.14	314	34.8	314	2365	1274
1000×20	11.67	48	9.2	48	71	126
1000×50	42.04	230	29.61	230	686	674
1000×100	111.6	608	71.6	608	–	Insufficient

The memory requirements for both the multifrontal UMFPACK 4.4 direct solver and GMRES solver (with preconditioner from the direct solver) are identical since both store the same *LU* factors. The only difference is that in the case of the direct solver, the *LU* factors are updated after each iteration, where as in the case of GMRES, the *LU* factors (and hence the preconditioner) are held fixed after the first iteration. In contrast, the memory requirement for the unifrontal solver is 2–3 times greater.

An important observation for the memory requirement of the unifrontal solvers is that as the number of elements in the *x*-direction increases, the memory requirement increases linearly. An increase of elements in the *y*-direction causes a quadratic increase in memory requirement. In the case of multifrontal solvers, an increase of elements in the *x*-direction causes a linear increase in the memory requirement and an increase in *y*-direction causes a superlinear increase in memory requirement. For problems involving large number of elements with almost equal number of elements in the *x*- and *y*-direction, the memory requirement of unifrontal solvers becomes prohibitively large while multifrontal solvers can accommodate reasonably large mesh sizes on a 2 GB RAM machine. For instance, mesh sizes of up to 400×400 can be run on a 2 GB RAM machine using multifrontal solvers for simple two-dimensional flow problems. Multifrontal solvers are therefore quite memory efficient for such applications.

Table 2 also shows the comparison of total CPU times for the three solvers. For the unifrontal solver, an increase in number of elements in the *x*-direction causes a proportionate increase in the total computational time, whereas an increase in the number of elements in the *y*-direction causes a huge increase in the total computational time. For the multifrontal solver, the trends in computation times are similar to those in memory requirements. The GMRES solver yields further reduction in the computational time over and above the multifrontal direct solver. The savings in computational time is less for coarse meshes, but with increase in fineness of mesh, especially with the increase in the number of elements in the *y*-direction, the savings in the computational time of GMRES solver can be large, making the solver computationally very attractive, with the same memory requirements as that of the direct solver.

As seen in Table 2, the memory requirement and computation time of a multifrontal solver is not only a function of the total number of nodes, but is also a function of the grid aspect ratio, *ar* (defined here as the ratio of the number of elements (*nex*) in the *x*-direction to the number of elements (*ney*) in the *y*-direction). A correlation for the memory requirement as a function of the total number of nodes and the grid aspect ratio is obtained by using a data set of 50 cases with different meshes. The following correlations are obtained for the two solvers:

$$M_1 = 4.96 \times 10^{-4} n^{1.278} ar^{-0.2}; \quad R^2 = 0.961 \tag{16}$$

$$T_1 = 1.6 \times 10^{-4} n^{1.22} ar^{-0.23}; \quad R^2 = 0.992 \tag{17}$$

$$T_2 = 1.5 \times 10^{-4} n^{1.16} ar^{-0.11}; \quad R^2 = 0.998 \tag{18}$$

where M_1 is the memory requirement of direct multifrontal solver (as well as the GMRES solver) in megabytes, T_1 is the CPU time (in seconds) of direct multifrontal solver, T_2 is the CPU time (in seconds) of GMRES solver, n is the total number of nodes in the mesh, ar is the grid aspect ratio (defined as the ratio of *nex* to *ney*) and R^2 is the regression coefficient. It is clear that for a given total number of nodes, as the aspect ratio increases, the computational time and memory decrease significantly. Note that the required memory is the same for the direct as well as GMRES solver since the preconditioner for the latter is based on the direct solver. The correlation indicates the superlinear increase in the memory requirement for these solvers. The correlations for time indicate steeper superlinear increase in the computational time for the direct solver (exponent 1.22 in Eq. 17) as compared to that for the GMRES solver (exponent 1.16 in Eq. 18). The computational time for GMRES solver is also not as strong a function of the grid aspect ratio as the direct solver. This shows the advantage of using a GMRES type solver for meshes with larger number of elements and for problems where the grid aspect ratio is close to unity. When the grid aspect ratio is close to unity, the multifrontal solver becomes inefficient, as is the characteristic of frontal routines. But the implementation of GMRES solver rectifies this problem and increases the efficiency in terms of CPU time.

5.2 Particulate flow with six size classes

The GMRES solver's efficiency is expected to be much more significant in multiphase problems. As a case study, multi-size particulate flow (with six size classes representing the particle size distribution) in a rotating channel is considered here. The detailed formulation of this problem with solutions using direct UMFPACK 4.4 solver has been presented elsewhere (Gupta and Pagalthivarthi, 2006). The solutions have been validated (Gupta and Pagalthivarthi, 2006) by mesh refinement and comparison with experimental results. Q_1Q_0 finite elements were used in the numerical computations.

Figures 3 and 4 are included to demonstrate the close agreement between the solutions obtained using the direct solver and the GMRES solver. Simultaneously, they show a comparison of the numerical predictions with experimental results. Figure 3 shows the comparison of fully developed profiles of predicted mixture velocity and overall concentration using multifrontal UMFPACK 4.4 direct solver and iterative (GMRES) solver with the experimental results of Kaushal et al. (2002). It is clear that the GMRES and multifrontal direct solvers yield practically identical results. The agreement with experimental results is also respectable. Figure 4 shows the comparison between experimentally obtained species concentration profiles for the six size classes with predicted results using both solvers. The experimental results correspond to the multi-size particulate flow in a stationary duct. By adjusting the rotation rate, a Coriolis acceleration equivalent to the gravitational acceleration in a stationary channel is obtained. A detailed discussion on this validation method is presented in Gupta and Pagalthivarthi (2006).

To compare the performance of the two solvers, solutions for several meshes were obtained for the case of $\mathrm{Re}_H = 1.375 \times 10^5$ (height-based bulk Reynolds number), $\mathrm{Ro}_H = 0.032$ (height-based rotation number) and $C_{\mathrm{avg,inlet}} = 12.64\%$ with channel length, $L = 4$ m and channel height, $H = 0.05$ m. The slurry has six representative particle size classes ranging from 38 μm to 738 μm. For a 600×48 mesh, the multifrontal direct UMFPACK 4.4 solver takes about 105 minutes, while the GMRES solver takes about 75 minutes for overall convergence of all the field variables. In comparison, since the unifrontal solver for this case takes about 15 hours for overall convergence, it is excluded in the subsequent discussion for multi-size particulate flow. An overall convergence (or final solution) is achieved when the correction vector in the infinity norm of all the field variables becomes less than a prescribed tolerance (set to 10^{-6} in the present study).

Figure 3 Comparison of experimental results (Kaushal et al., 2002) with predicted (a) overall concentration; (b) mixture velocity profiles; (c) difference in concentration and (d) mixture velocity profiles obtained from GMRES solver and UMFPACK 4.4 direct solver

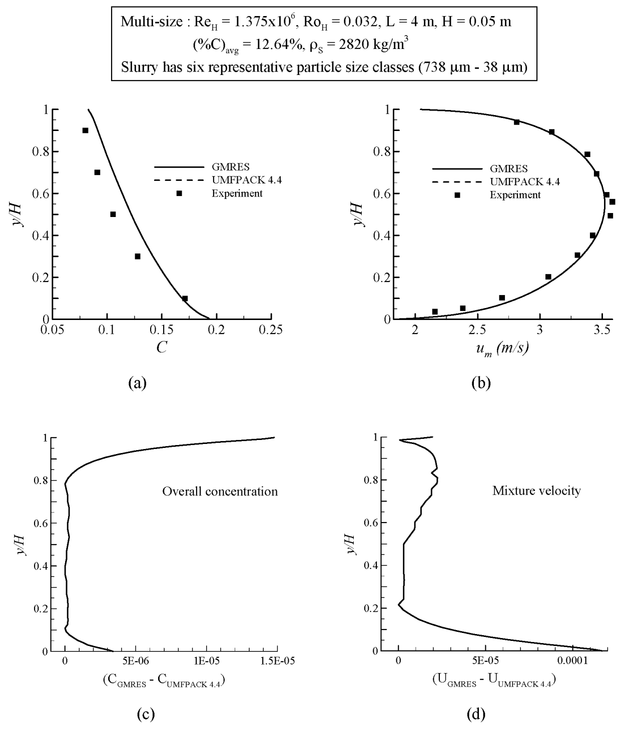
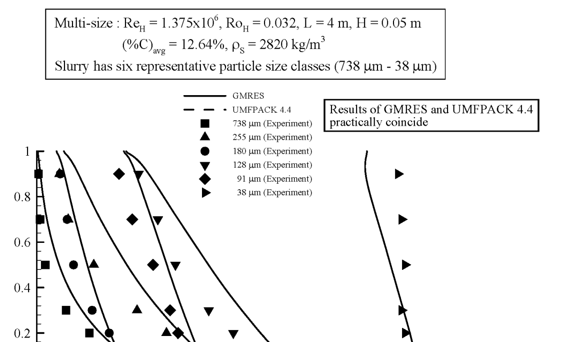


Figure 4 Comparison of experimental results (Kaushal et al., 2002) with predicted individual species concentration profiles obtained using GMRES solver and UMFPACK 4.4 direct solver



In Table 3, a comparison of the total CPU time (in hours) and memory requirement for the multi-size particulate flow through rotating channel is shown for several meshes. The results indicate that even for coarser meshes, the computational time of GMRES is significantly less compared to that of the direct solver. But as the mesh fineness increases in the *y*-direction, a significant reduction in computational time is observed. For example, for a 300×300 mesh, the computational time for the direct solver is approximately 32 hours and for the GMRES solver (using direct solver as preconditioner), it is approximately 12 hours. Thus, a reduction of over 60% of the computational time is seen compared to that of the direct solver. For 400×400 mesh, the direct solver takes approximately six days, while the GMRES solver takes approximately two days. Hence, there is significant savings in computational time. For large-scale problems and especially as the number of elements in the *y*-direction increases (keeping *nex* constant), GMRES is found to be significantly more efficient compared to the direct (UMFPACK 4.4) solver. The main limitation is the memory requirement. Since GMRES is based on the preconditioner generated from the direct solver, the memory requirement of GMRES is the same as that of the direct solver (as in the case of the simple laminar flow problem considered in preceding discussion). Indeed, the memory requirement for the two solvers is given by Equation (16) as before. This is because the critical memory-determining step in particulate flow solution is the solution of the mixture velocity and pressure (u_m, v_m, p) by combined Newton's iteration.

Table 3 Comparison of total CPU time (in hours) for the computation of multi-size rotating channel flow using both UMFPACK 4.4 and GMRES solver

	Computation time (hours)		
Mesh size	*UMFPACK 4.4* (*Direct Multifrontal*)	*GMRES*	*Ratio* (*GMRES/Direct*)
300×24	0.31	0.23	0.728
300×48	0.83	0.58	0.7
300×64	2.48	1.47	0.59
300×100	3.37	1.91	0.56
300×300	31.89	11.79	0.37
600×48	1.69	1.22	0.71
600×64	3.04	1.89	0.62
600×100	6.63	3.87	0.58
800×48	2.45	1.67	0.68
1000×48	3.01	2.14	0.707
400×400	149.5	45.33	0.3
600×300	74.4	30.0	0.4

Typically the memory requirement of a mesh with 200,000 elements is 4 GB. So in this study, meshes finer than 600×300 could not be used since the computations were performed on an Intel® Xeon Dual Processor 2.4 MHz machine with 4 GB of RAM. With the availability of machines with larger RAM capacity, these solvers can be tested for finer meshes like 600 × 600, 1000 × 1000 and so on. Based on the present trend, GMRES could be predicted to be as fast as five times or more compared to the direct multifrontal solver and it could save computational time in terms of days for such high-demand applications.

Table 4 gives a detailed insight of the performance of both the solvers while solving different sets of non-linear equations. The different sets of non-linear equations can be broadly classified into three categories:

- mixture flow field equations solving effectively three variables (u_m, v_m at each node and p for each element)
- solid velocity flow field equations for a particular species solving two variables (u_S, v_S at each node)
- concentration flow field for a particular species solving one variable C at each node (i.e., one degree of freedom at each node).

Table 4 Comparison of memory requirements and CPU time/iteration for the solution of different sets of flow field equations using both solvers

			Computation time (sec)		
Flow field equations	*Mesh size*	*Memory* (MB)	*UMFPACK 4.4* (*Direct Multifrontal*)	*GMRES*	*Ratio* (*GMRES/Direct*)
Mixture flow field equations	300 × 24	23.7	1.32	0.54	0.40
	300 × 48	66.7	3.4	1.3	0.38
	300 × 64	98	4.6	1.7	0.37
	300 × 100	181	8.5	2.62	0.30
	300 × 300	828	45.3	9	0.20
	600 × 48	132	6.71	2.5	0.37
	600 × 64	191	10	3.5	0.35
	600 × 100	355	17.3	5.3	0.30
	600 × 300	1717	123	21	0.17
	400 × 400	1605	115	17	0.15
Solid velocity field equations	300 × 24	11	0.86	0.55	0.64
	300 × 48	27	1.88	1.15	0.61
	300 × 64	41	2.66	1.54	0.58
	300 × 100	74	4.6	2.5	0.54
	300 × 300	281	16.6	7.8	0.47
	600 × 48	56	3.77	2.3	0.61
	600 × 64	82	5.35	3.04	0.57
	600 × 100	143	8.69	4.7	0.54
	600 × 300	553	38	18	0.47
	400 × 400	508	32.6	15	0.46
Concentration field equations	300 × 24	3.4	0.4	0.3	0.75
	300 × 48	8	0.82	0.59	0.72
	300 × 64	12	1.09	0.79	0.72
	300 × 100	21	1.85	1.22	0.66
	300 × 300	77	6.4	3.9	0.61
	600 × 48	16	1.7	1.2	0.70
	600 × 64	24	2.32	1.65	0.71
	600 × 100	40	3.55	2.44	0.69
	600 × 300	151	13.1	7.7	0.59
	400 × 400	144	12	7	0.58

The nature of the global matrix structure for these three sets of equations is different from each other, leading to a difference in the performance of the GMRES solver in comparison to the multifrontal direct solver. The memory requirements and the computational times for these three sets of equations are explained as follows.

From Table 4, it is observed that the memory requirement for solid velocity equations is almost three times that of the concentration flow field equations; and that of the mixture flow field equations is almost seven to eight times that of the concentration flow field equations. The more the number of degrees of freedoms that is solved at each node, the memory requirement increases superlinearly. This is expected because the connectivity between the degrees of freedoms as observed on the computational grid increases. This leads to more amount of fill-in (non-zero entries) in the *LU* factors in addition to the increase in number of equations. The increased fill-in results in a larger memory requirement for the storage of *LU* factors. This explains the rather high memory requirements and computational time for mixture flow field equations.

Table 4 also shows the comparison of the time taken by GMRES solver for these three different sets of equations. The average CPU time taken per iteration for both UMFPACK 4.4 and GMRES solver is presented. As discussed previously, the memory requirements of the mixture flow field equations is much higher compared to the other sets of equations. This implies that the factorisation time (computation of the non-zeroes of the *LU* factors) is correspondingly higher. Since GMRES skips the factorisation step (starting from the second iteration), its performance is found to be the best in the case of mixture flow equations. Especially for the case of very fine meshes with nearly equal number of elements in the *x*- and *y*- directions, the savings in computational time in mixture flow field equations is huge (for a 300 × 300 mesh, GMRES iterations take only 20% of the computational time taken by the direct solver). The trend in the performance of GMRES solver within each set of equations is similar with respect to the mesh fineness. With the increase in the number of elements in the *x*-direction, the performance of GMRES in comparison to the direct solver is not much affected. With the increase in the number of elements in the *y*-direction, the GMRES solver performs more efficiently in comparison to the direct solver. The explanation for this behaviour is the superlinear increase in the factorisation time (time for computation of *LU* factors), when the elements in the *y*-direction is increased, whereas the time for GMRES solver increases linearly.

The overall computation time for converged multi-size particulate flow solutions using the direct and GMRES solvers have been determined for several meshes including those shown in Table 3. Correlations for the overall computation time are found to be

$$T_1 = 0.019 n^{1.4} ar^{-0.51}; \quad R^2 = 0.91 \tag{19}$$

$$T_2 = 0.12 n^{1.12} ar^{-0.42}; \quad R^2 = 0.923 \tag{20}$$

where T_1 is the total CPU time (in seconds) for the direct multifrontal solver, T_2 is the total CPU time (in seconds) of the GMRES solver, n is the total number of nodes and ar is the grid aspect ratio.

An important concern for using a multifrontal solver (either directly or as preconditioner for GMRES) is its huge memory requirement. In 3D problems, this can seriously inhibit the use of direct solvers. The feasibility of extending the present 2D computations to 3D is examined next with the example of non-linear heat conduction in three dimensions.

5.3 Three dimensional heat conduction

The reason for choosing this problem is to get a quick estimate of the memory requirements of direct solver for 3D computation. By choosing a non-linear problem, the performance of the present GMRES implementation based on multifrontal preconditioner can be compared with the performance of the standard ILU based solvers for 3D computations.

ILU solvers are based on incomplete Gaussian elimination method with levels of fill-in chosen according to a predefined strategy. The ILU solver used here is the ILU(*k*), where the level of fill-in (*k*) is chosen as 15, to ensure convergence for all grid sizes considered here. As the level of fill-in increases, the robustness of the ILU solver increases, but the memory requirement for the storage of *LU* factors also increases.

Table 5 shows the computation time and memory requirements for UMFPACK 4.4 (multifrontal direct solver), GMRES (UMFPACK based) and ILU(15). For multifrontal direct solver-based solutions, the memory requirement is also a function of the distribution elements in the three directions, expressed as aspect ratios in the *x-y* ($ar_1 = nex/ney$) and in the *x-z* ($ar_2 = nex/nez$) planes.

Table 5 shows that multifrontal direct solver becomes expensive for 3D problems both in terms of memory requirement and the computational time as the total number of nodes increase. The following correlations are obtained.

$$M_1 = 6.8 \times 10^{-6} n^{1.7} ar_1^{-0.3} ar_2^{-0.3}; \quad R^2 = 0.91 \tag{21}$$

$$M_2 = 7.2 \times 10^{-4} n^{1.076} ar_1^{-0.035} ar_2^{-0.035}; \quad R^2 = 0.99 \tag{22}$$

where M_1 is the memory requirement of multifrontal solver (as well as GMRES) in megabytes, M_2 is the memory requirement of ILU(15) solver, n is the total number of nodes and ar_1 and ar_2 are the grid aspect ratio's as defined above.

The ILU solver's memory requirement is nearly independent of the aspect ratio and nearly linearly dependent on the total number of nodes. On the other hand, the multifrontal and GMRES solvers' memory requirement depends superlinearly on the total number of nodes. It also depends on the aspect ratios significantly; as the number of elements in one direction increases in comparison to the other directions (keeping n constant), the memory requirement decreases significantly.

Table 5 Comparison of memory requirements and CPU time for the using UMFPACK 4.4, GMRES (UMFPACK-based) and ILU solvers (partial list of runs)

				Memory requirement (MB)		*Total CPU time* (seconds)		
nex	*ney*	*nez*	*Total nodes*	*Direct multifrontal or GMRES*	*ILU(15)*	*Direct multifrontal*	*GMRES*	*ILU(15)*
50	10	10	6171	8.07	7.64	2.56	1.96	2.625
100	10	10	12221	18.25	15.42	5.17	3.86	5.98
200	10	10	24321	36.58	31.01	10.46	7.73	12.82
400	10	10	48521	72.29	61.74	21.68	15.8	23.95
600	10	10	72721	107.67	89.83	33.15	24.82	33.14
50	20	10	11781	25.76	15.9	6.53	4.28	5.61
100	20	10	23331	54.2	32.1	14.11	8.67	12.25
200	20	10	46431	100.72	69.68	27.86	17.23	26.48
400	20	10	92631	191.8	127.7	56.26	35.4	42.3
600	20	10	138831	306.2	185.4	86.26	54.3	69.5
100	30	10	34441	112	49.64	28.94	14.78	20.21
200	30	10	68541	187.72	99.84	55.21	27.31	43.51
400	30	10	136741	420	199.54	168.2	67.2	90.25
500	30	10	170841	540	248.61	243.2	71.3	107.92
50	20	20	22491	132	33.92	32.75	12.48	11.44
100	20	20	44541	299.4	68.78	87.2	20.32	26.23
200	20	20	88641	536.73	138.16	178.2	54.2	62.4
300	20	20	132741	726.2	207.57	273.2	68.3	81.2
400	20	20	176841	1107	276.89	381.2	92.45	121.1
20	20	20	9261	42.17	13.1	8.8	3.8	3.75
30	30	30	29791	208.7	46.66	53.2	15.3	14.34
40	40	40	68921	870.75	113	411.5	42.3	37.2
46	46	46	103823	1398.43	174.6	1033.4	71.3	60
50	50	50	132651	2967.56	225	1875.4	101.2	75
30	20	20	13671	76.58	20.02	16.83	6.86	6.18
30	30	20	20181	102.1	30.5	27.76	11.12	9.861
40	20	20	18081	96.14	26.96	24.9	9.765	8.656
40	30	20	26691	156.68	41.06	39.25	14.33	13.91
50	20	20	22491	130.34	33.92	32.1	13.1	11.38
50	30	20	33201	197.65	51.63	52.12	17.3	17.97
50	30	30	49011	382.1	78.96	82.67	28.32	26.91
100	50	20	108171	1092.9	175.98	517.2	49.81	79.96
100	40	30	128371	1450.4	214.83	861.5	71.32	89.42
100	50	30	159681	1823.45	269.46	972.1	91.2	115.9
200	50	10	112761	521.2	168.85	152.14	34.2	76.48

The correlations for the total computation time in seconds required for the solution of the non-linear conduction problem for the three solvers in terms of the total number of nodes and grid aspect ratios are given as

$$T_1 = 2.5\times10^{-6} n^{1.7} ar_1^{-0.35} ar_2^{-0.32}; \quad R^2 = 0.968 \tag{23}$$

$$T_2 = 1.1\times10^{-5} n^{1.17} ar_1^{-0.12} ar_2^{-0.18}; \quad R^2 = 0.986 \tag{24}$$

$$T_3 = 3\times10^{-4} n^{1.1} ar_1^{-0.08} ar_2^{-0.12}; \quad R^2 = 0.99 \tag{25}$$

where T_1 is the total CPU time for direct multifrontal solver, T_2 is the total CPU time for GMRES solver and T_3 is the total CPU time for ILU solver.

For meshes with equal distribution of elements in the three directions, ILU(15) performs much better than GMRES solver. However, GMRES puts up a strong competition to ILU(15) for most of the cases involving aspect ratios much greater than unity. This indicates that GMRES performs well even for 3D problems when the aspect ratios of the number of elements in different directions are greater than unity. There are many flow problems with a dominant flow direction where this advantage could be profitably used. Flows through, channels, pipes, impellers and so forth are suitable examples.

The problem dealt with here has temperature as the only field variable. The multifrontal-based GMRES solver could be expected to perform attractively for 3D Navier-Stokes problems (with a dominant flow direction) by means of the fully coupled (velocity-pressure) combined Newton's algorithm with its quadratic convergence property. The standard ILU(k) implementation, on the other hand, does not work very well with fully coupled formulations because of the zeros arising from the discretisation of the continuity equation. To overcome this limitation, ILU(k) solvers are applied with segregated algorithms in combination with pressure projection techniques. This procedure loses the quadratic convergence property of Newton's method. Thus, GMRES with multifrontal-based preconditioner provides a good alternative to explore for 3D flow problems as well.

The main limitation in GMRES implementation (with direct solver preconditioner) is that of high memory. The memory requirement shoots up especially as the grid aspect ratios tends to unity. The memory requirement for 3D problems can be addressed by implementing the codes on 64-bit machines with larger RAM specifications. The multifrontal solvers are also known for their good scalability on distributed parallel machines. One such parallel implementation of the multifrontal solver is the MUltifrontal Massively Parallel Solver (MUMPS) packages developed by CERFACS group in France. The implementation of MUMPS solver on 64-bit machines can significantly increase the capability of using multifrontal-based GMRES solvers for 3D flow problems.

6 Conclusions

The study compares the memory requirement and computation time of direct multifrontal and iterative GMRES solvers for two-dimensional Navier-Stokes and multi-size particulate flow problems. The feasibility of applying GMRES to 3D problems is considered by comparing its performance with that of an ILU(k) solver for non-linear heat conduction problems. Correlations for computation time and required memory are presented for the different solvers used. The following conclusions are drawn based on the present study.

- Both the multifrontal UMFPACK 4.4 direct solver and the GMRES solver outperform the unifrontal solver (for laminar channel flow) by several times – both in terms of memory requirement and speed. The difference in the performance increases greatly as the ratio of the number of elements in the x-direction to that in the y-direction decreases.

  For the laminar channel flow problem, the computation time of GMRES solver is about 60% of the computation time of the multifrontal direct solver.

  For the multi-size particulate flow problem, the computation time of GMRES solver (with preconditioner obtained from direct solver) is about 30–60% (depending on the mesh) of the computation time of the multifrontal direct solver.

  The computational advantage of the GMRES solver relative to the multifrontal direct solver is seen to increase with

  - increasing total number of nodes
  - aspect ratio of the mesh (ratio of the number elements in the two directions) approaching unity.

  With an increase in the degrees of freedom to be solved at each node, the performance of preconditioned GMRES is greatly enhanced. Thus, maximum gains are realised in the computation of mixture velocity and pressure fields.

  Multifrontal-based GMRES is found to perform well even for 3D non-linear heat conduction problem compared to the ILU solvers when the ratio of the number of elements in different coordinate directions is about 2.5 or greater. Thus, there is good scope of extending the GMRES solver for 3D Navier-Stokes problems. Memory is clearly a limitation for the multifrontal-based GMRES solver.

Acknowledgement

This investigation was carried out at Indian Institute of Technology, Delhi, as part of the doctoral dissertation of Pankaj K. Gupta under the guidance of Dr. Krishnan V. Pagalthivarthi.

References

Adams, M.F. (1999) 'Parallel multigrid solver algorithms and implementation for 3D unstructured finite element problem', *Supercomputing '99*, November 13–19, Portland, Oregon.

Amestoy, P.R. and Duff, I.S. (1989) 'Vectorization of a multiprocessor multifrontal code', *Int. J. Supercomputer Appl.*, Vol. 3, No. 3, pp.41–59.

Brandt, A. (1977) 'Multi-level adaptive solutions to boundary value problems', *Math. Comp.,* Vol. 31, No. 8, pp.333–390.

Davis, T.A. (2004a) 'A column pre-ordering strategy for the unsymmetric-pattern multi-frontal method', *ACM Trans. Math. Software*, Vol. 30, No. 2, pp.165–195.

Davis, T.A. (2004b) 'Algorithm 832: UMFPACK – an unsymmetric-pattern multifrontal method', *ACM Trans. Math. Software*, Vol. 30, No. 2, pp.196–199.

Davis, T.A., Amestoy, P.R. and Duff, I.S. (2004a) 'Algorithm 837: AMD, an approximate minimum degree ordering algorithm', *ACM Trans. Math. Software*, Vol. 30, No. 3, pp.381–388.

Davis, T.A. and Duff, I.S. (1997) 'A combined unifrontal/multifrontal method for unsymmetric sparse matrices', *ACM Trans. Math. Software*, Vol. 25, No. 1, pp.1–19.

Davis, T.A., Gilbert, J.R. and Larimore, E. (2004) 'Algorithm 836: COLAMD, an approximate column minimum degree ordering algorithm', *ACM Trans. Math. Software*, Vol. 30, No. 3, pp.377–380.

Einset, E.O. and Jensen, K.F. (1992) 'A finite element solution of three-dimensional mixed convection gas flows in horizontal channels using preconditioned iterative matrix methods', *International Journal for Numerical Methods in Fluids*, Vol. 14, No. 7, pp.817–841.

Ghia, U., Ghia, K.N. and Shin, C.T. (1982) 'High-Re solutions for incompressible flow using the Navier-Stokes equations and multigrid method', *J. Comput. Phys.,* Vol. 48, No. 3, pp.387–411.

Glowinski, R., Pan, T.W., Hesla, T.I. and Joseph, D.D. (1999) 'A distributed Lagrange multiplier/fictitious domain method for particulate flows', *Int. J. Multiphase Flow*, Vol. 25, No. 5, pp.755–794.

Gupta, P.K. and Pagalthivarthi, K.V. (2006) 'Finite element prediction and validation of multi-size particulate flow in rotating channels', in print (as of December 2006) *Progress in Computational Fluid Dynamics*.

Irons, B.M. (1970) 'A frontal solution scheme for finite element analysis', *Numer. Meth. Engg.*, Vol. 2, pp.5–32.

Kaushal, D.R., Seshadri, V. and Singh, S.N. (2002) 'Prediction of concentration and particle size distribution in the flow of multi-sized particulate slurry through rectangular duct', *Appl. Math. Model.,* Vol. 26, No. 10, pp.941–952.

Kurreck, M. and Wittig, S. (1997) 'A comparative study of pressure correction and block-implicit finite volume algorithms on parallel computers', *Int. J. Num. Meth. in Fluids*, Vol. 24, No. 11, pp.1111–1128.

Pagalthivarthi, K.V. and Ramanathan, V. (2002) 'Finite element study of free surface turbulent flow in rotating channels', *Int. J. Numerical Meth. Fluids*, Vol. 38, No. 3, pp.283–305.

Pagalthivarthi, K.V., Ravichandra, J.S. and Sanghi, S. (2005) 'Multi-size particulate flow in horizontal ducts: modelling and validation', *Progress in Computational Fluid Dynamics*, Vol. 5, No. 8, pp.466–481.

Patankar, N.A. and Joseph, D.D. (2001) 'Lagrangian numerical simulation of particulate flows', *Int. J. Multiphase Flow*, Vol. 27, No. 10, pp.1685–1706.

Phillips, R.E. and Schmidt, F.W. (1985) 'A multilevel multigrid technique for recirculating flows', *Numerical Heat Tranfer*, Vol. 8, No. 5, pp.573–594.

Reddy, J.N. and Gartling, D.K. (2001) *The Finite Element Method in Heat Transfer and Fluid Dynamics*, 2nd ed., CRC Press, Florida.

Saad, Y. (2003) *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM Publications, Philadelphia.

Saad, Y. and Schultz, M.H. (1986) 'GMRES: a generalized minimal residual algorithm for solving non-symmetric linear systems', *SIAM J. Sci. Stat. Comput.*, Vol. 7, No. 3, pp.856–869.

Schlichting, H. (1979) *Boundary Layer Theory*, in Kestin, J. (Trans.): McGraw-Hill Book Company, New York.

Slater, S.A., Leeming, A.D. and Young, J.B. (2003) 'Particle deposition from two-dimensional turbulent gas flows', *Int. J. Multiphase Flow*, Vol. 29, No. 5, pp.721–750.

Slater, S.A. and Young, J.B. (2001) 'The calculation of inertial particle transport in dilute gas-particle flows', *Int. J. Multiphase Flow*, Vol. 27, No. 1, pp.61–87.

Sommerfeld, M. and Huber, N. (1999) 'Experimental analysis and modeling of particle-wall collisions', *Int. J. Multiphase Flow*, Vol. 25, Nos. 6–7, pp.1457–1489.

Vandervorst, H.A. and Vuik, C. (1994) 'GMRESR: a family of nested GMRES methods', *Numerical Linear Algebra with Applications*, Vol. 1, No. 4, pp.369–386.

Vanka, S.P. (1985) 'Block-implicit calculation of steady turbulent recirculating fluid flows', *International J. Heat Mass Transfer*, Vol. 28, No. 11, pp.2093–2103.

Vanka, S.P. (1986) 'A calculation procedure for three-dimensional steady recirculating flows using multigrid methods', *Computer Methods in Applied Mechanics and Engineering*, Vol. 55, No. 3, pp.321–338.

Appendix

GMRESR is a recursive variant of GMRES algorithm. In this algorithm, two iterative loops are performed – inner loop and an outer loop, both based on GMRES algorithm. The idea behind using a recursive version of GMRES is to apply an internal preconditioning operation using GMRES itself. The inner loop performs this preconditioning operation using a few GMRES steps. Details of its implementation and the practical issues involved are discussed in Vandervorst and Vuik (1994). The GMRESR algorithm with preconditioner obtained from the multifrontal direct solver is presented here:

```
(1) Select $x_0$, $m$, $tol$;
      $r_0 = b - Ax_0, k = -1;$
      $r_0 = M^{-1}r_0,$
      ($M$ is the preconditioner obtained from direct solver)
(2) do until $\| r_{k+1} \|_2 > tol$
        $k = k + 1;$
        $w_k = M^{-1}r_k;$
        Solve $Au_k^{(0)} = w_k$ (use $m$ iterative steps of GMRES (Saad and Schultz, 1986));
        $c_k^{(0)} = Au_k^{(0)};$
```

```
for $i = \max(0, k - j), \ldots, k - 1$
        $\alpha_i = c_i^T c_k^{(i)};$
        $c_k^{(i+1)} = c_k^{(i)} - \alpha_i c_i;$
        $u_k^{(i+1)} = u_k^{(i)} - \alpha_i c_i;$
$c_k = c_k^{(k)} / \| c_k^{(k)} \|_2 ; u_k = u_k^{(k)} / \| c_k^{(k)} \|_2 ;$
        $x_{k+1} = x_k + u_k c_k^T r_k;$
        $r_{k+1} = r_k - c_k c_k^T r_k;$
```

In the above preconditioned GMRESR algorithm, x_0 is the initial guess, m is the number of inner GMRES iterations, *tol* is the tolerance limit, r is the residual vector, u is an auxiliary vector, c is also an auxiliary vector satisfying the orthogonality conditions ($r_{k+1} \perp c_k$ and $c_i \perp c_j$ for $i \neq j$), and α is an auxiliary scalar.