A Parallel PSPG Finite Element Method for Direct Simulation of Incompressible Flow

Jörg Stiller[1], Karel Fraňa[1], Roger Grundmann[1],
Uwe Fladrich[2], and Wolfgang E. Nagel[2]

[1] Institute for Aerospace Engineering, TU Dresden, D-01062 Dresden, Germany
{stiller,frana,grundmann}@tfd.mw.tu-dresden.de
[2] Center for High Performance Computing, TU Dresden, D-01062 Dresden, Germany
{fladrich,nagel}@zhr.tu-dresden.de

Abstract. We describe a consistent splitting approach to the pressure-stabilized Petrov-Galerkin finite element method for incompressible flow. The splitting leads to (almost) explicit predictor and corrector steps linked by an implicit pressure equation which can be solved very efficiently. The overall second-order convergence is proved in numerical experiments. Furthermore, the parallel implementation of the method is discussed and its scalability for up to 120 processors of a SGI Origin 3800 system is demonstrated. A significant superlinear speedup is observed and can be attributed to cache effects. First applications to large-scale fluid dynamic problems are reported.

1 Introduction

We are interested in direct numerical simulations (DNS) of transitional and turbulent flows. Traditionally, specialized finite difference or spectral methods are used for this purpose. Though very efficient, these methods are often restricted to simple configurations. Unstructured finite volume methods and finite element methods are more flexible and offer the potential benefit of easier incorporating adaptive techniques. On the other hand, they are computationally less efficient and more difficult to parallelize. Also, the discretization scheme has to be carefully designed to meet the accuracy requirements for DNS.

In this paper, we consider a pressure-stabilized Petrov/Galerkin finite element method (PSPG-FEM) based on linear shape functions [1]. In Section 2, we describe a splitting approach that is similar to common projection and fractional step methods (see, e.g. [2]) but novel in the context of PSPG-FEM. The splitting yields an implicit Poisson-type equation for the pressure and an almost explicit predictor-corrector scheme for the velocity. In Section 3, we discuss the implementation on top of our in-house MG grid library [3]. Numerical accuracy and scalability of the method are examined in Section 4. In Section 5, we briefly discuss the application to DNS of electromagnetic stirring with rotating magnetic fields.

M. Danelutto, D. Laforenza, M. Vanneschi (Eds.): Euro-Par 2004, LNCS 3149, pp. 726–733, 2004.

2 Finite Element Method

The flows under consideration are governed by the incompressible Navier-Stokes equations

$$\partial_t \boldsymbol{u} + \nabla \cdot \boldsymbol{u}\boldsymbol{u} = -\nabla p + \nu \nabla^2 \boldsymbol{u} + \boldsymbol{f}$$
$$\nabla \cdot \boldsymbol{u} = 0$$

where $\boldsymbol{u}$ is the velocity, p is the pressure divided by density, ν is the kinematic viscosity and $\boldsymbol{f}$ is the body force per unit mass. After triangulating the computational domain Ω into elements $\{\Omega_e\}$, the pressure-stabilized Petrov/Galerkin formulation of the problem can be stated as

$$\int_\Omega \big[\boldsymbol{w} \cdot (\partial_t \boldsymbol{u} + \nabla \cdot \boldsymbol{u}\boldsymbol{u} + \nabla p - \boldsymbol{f}) + (\nabla \boldsymbol{w})^{\mathrm{T}} : \nu \nabla \boldsymbol{u}\big] \mathrm{d}\Omega = \int_\Gamma \boldsymbol{w} \cdot \nu \partial_n \boldsymbol{u} \, \mathrm{d}\Gamma$$
$$\int_\Omega q \nabla \cdot \boldsymbol{u} \, \mathrm{d}\Omega + \sum_e \int_{\Omega_e} \tau \nabla q \cdot \boldsymbol{r}(\boldsymbol{u}, p) \, \mathrm{d}\Omega = 0$$

where $\boldsymbol{w}$ and q are the momentum and continuity weight functions, Γ is the boundary with normal $\boldsymbol{n}$, τ is the elementwise defined stabilization coefficient, and

$$\boldsymbol{r}(\boldsymbol{u}, p) = \partial_t \boldsymbol{u} + \nabla \cdot \boldsymbol{u}\boldsymbol{u} + \nabla p - \nu \nabla^2 \boldsymbol{u} - \boldsymbol{f} = \boldsymbol{r}^*(\boldsymbol{u}) + \nabla p$$

is the momentum residual. Dropping the stabilization term ($\tau = 0$) recovers the standard Galerkin formulation which, however, is unstable for equal-order interpolations. Appropriate choices for τ are given in [4]. In the following we use $\tau =$ constant, for simplicity.

The PSPG formulation can be formally integrated in time to give

$$\int_\Omega \left[\boldsymbol{w} \cdot \Big(\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} + \nabla \cdot \overline{\boldsymbol{u}\boldsymbol{u}} + \nabla \bar{p} - \bar{\boldsymbol{f}}\Big) + (\nabla \boldsymbol{w})^{\mathrm{T}} : \nu \nabla \bar{\boldsymbol{u}}\right] \mathrm{d}\Omega = \int_\Gamma \boldsymbol{w} \cdot \nu \partial_n \bar{\boldsymbol{u}} \, \mathrm{d}\Gamma$$
$$\int_\Omega q \nabla \cdot \boldsymbol{u}^{n+1} \, \mathrm{d}\Omega + \sum_e \int_{\Omega_e} \tau \nabla q \cdot \bar{\boldsymbol{r}}(\boldsymbol{u}, p) \, \mathrm{d}\Omega = 0$$

Here, the overbar denotes the average over the time interval (t_n, t_{n+1}). Introducing the elementwise polynomial approximation yields the discrete equations which, symbolically, can be stated as

$$M \frac{\boldsymbol{U}^{n+1} - \boldsymbol{U}^n}{\Delta t} + \bar{\boldsymbol{N}} + \boldsymbol{D}\bar{P} + \nu L \bar{\boldsymbol{U}} - \bar{\boldsymbol{F}} = -\nu L^\Gamma \bar{\boldsymbol{U}}$$
$$\boldsymbol{D} \cdot \boldsymbol{U}^{n+1} + \tau(\boldsymbol{D}^{\mathrm{T}} \cdot \bar{\boldsymbol{R}}^* + L\bar{P}) = 0$$

where $\boldsymbol{U}$, P, $\boldsymbol{F}$ and $\boldsymbol{R}$ represent the expansion coefficients, M is the mass matrix, $\bar{\boldsymbol{N}}$ the contribution of the nonlinear term, $\boldsymbol{D}$ is the gradient matrix, L is the Laplace matrix and L^Γ the related boundary contribution. The momentum equation can be split into a velocity (predictor) and a pressure (corrector) step:

$$\boldsymbol{U}^* = \boldsymbol{U}^n - \Delta t M^{-1}(\bar{\boldsymbol{N}} + \nu(L + L^\Gamma)\bar{\boldsymbol{U}} - \bar{\boldsymbol{F}})$$

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^* - \Delta t M^{-1} \boldsymbol{D} \bar{P}$$

Employing $\bar{\boldsymbol{R}}^* = (\boldsymbol{U}^* - \boldsymbol{U}^n)/\Delta t$, the following pressure equation can be obtained:

$$[(1-\delta)\boldsymbol{D} \cdot M^{-1}\boldsymbol{D} + \delta L]\bar{P} = -\boldsymbol{D} \cdot \boldsymbol{U}^*/\Delta t$$

where $\delta = \tau/\Delta t$. It is worth noting that the last three equations are still consistent with the exact solution of the continuous problem, if $\bar{\boldsymbol{N}}$ is properly defined. We further remark that $\delta = 1$ is related to projection methods using the continuous pressure equation, and $\delta = 0$ corresponds to the standard Galerkin method.

In our piecewise linear approximation, the nonlinear term is evaluated as

$$\bar{\boldsymbol{N}} = \boldsymbol{D} \cdot \bar{\boldsymbol{U}}\bar{\boldsymbol{U}}$$

where $\bar{\boldsymbol{U}}\bar{\boldsymbol{U}}$ represents the nodal values of the momentum flux tensor $\bar{\boldsymbol{u}}\bar{\boldsymbol{u}}$. The time averages are evaluated using the second-order Adams-Bashforth method. Furthermore, the pressure gradient is modified using the Gresho-Chan trick [5]. The resulting, final splitting scheme is

$$\boldsymbol{U}^* = \boldsymbol{U}^n - \Delta t M^{-1}(\boldsymbol{D} \cdot \bar{\boldsymbol{U}}\bar{\boldsymbol{U}} + \nu(L + L^{\Gamma})\bar{\boldsymbol{U}} - \bar{\boldsymbol{F}})$$

$$[(1-\delta)\boldsymbol{D} \cdot M_{\mathrm{L}}^{-1}\boldsymbol{D} + \delta L]\bar{P} = -\boldsymbol{D} \cdot \boldsymbol{U}^*/\Delta t$$

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^* - \Delta t M_{\mathrm{L}}^{-1} \boldsymbol{D} \bar{P}$$

where M_{L} denotes the lumped mass matrix.

3 Parallel Implementation

The numerical model was implemented on top of the MG grid library. MG provides data structures and procedures for handling unstructured grids as well as stable and fast methods for grid adaptation [3, 6]. Both, the MG library and the flow solver are coded in Fortran 95 and use MPI for communication.

Parallelization with MG is based on grid partitioning. In the simple case considered here, the grid generated by an external grid generator is decomposed into a specified number of partitions using the MeTiS package [7]. For adaptive simulations, a recursive multilevel-partitioning strategy is available [3].

In each time step, the numerical model requires the solution of two linear algebraic systems. In the predictor step, the consistent finite element mass matrix must be resolved. Since this matrix is well conditioned, a few (1–3) damped Jacobi iterations are sufficient. The pressure equation is solved using the CG method. As a good initial approximation is available from the preceding time step, usually 10–100 CG iterations provide the new pressure with the required accuracy. Alternatively, a multigrid method using a damped Jacobi smoother and a CG coarse grid solver is under consideration. It may be expected that the multigrid solver is more effective in large-scale simulations, but this has no been verified yet.

Basically, the execution of one time step essentially involves two types of global operations which have to be performed in parallel:

– the computation of matrix-vector products
– the scalar product of coefficient vectors

The implementation of the latter is trivial, as only the contribution of multiple copies (in shared grid nodes) has to be canceled. The matrix-vector products can be expressed as, e.g.,

$$L_{ij}\bar{P}_j = \sum_p L_{ij}^p \bar{P}_j$$

where L_{ij}^p is the contribution from the elements of partition p. These local matrices are precomputed and stored in an edge-based data basis. Once the local products are evaluated, the final result is obtained by adding the individual contributions in all nodes shared between one or more partitions. In the actual implementation, this operation is realized just by calling the appropriate procedure of the MG library.

4 Accuracy and Performance

Two laminar flow configurations with known analytic solution were used to examine the accuracy of the method: The transient channel flow between two parallel plates, and the stationary flow of an electro-conducting fluid in an infinite cylinder that is exposed to a rotating magnetic field (see [8]). Unstructured tetrahedral grids with different mesh spacing h were used in both cases. The results shown in Fig. 1 clearly reveal a convergence rate of order 2.

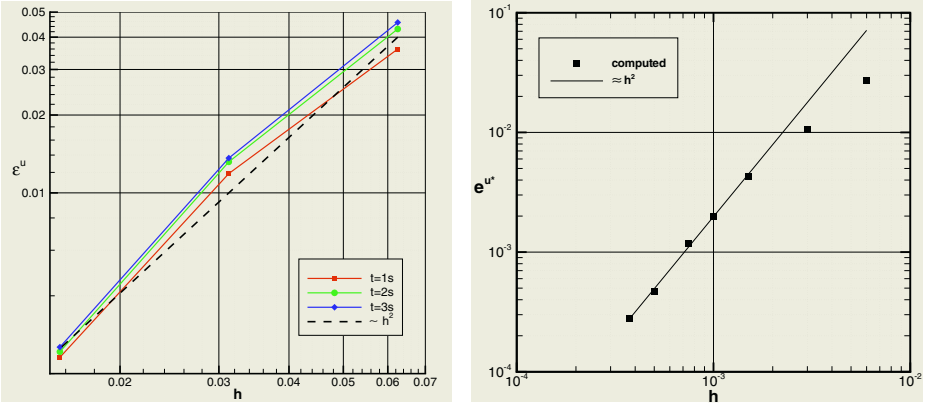

Fig. 1. L_∞ velocity error in transient channel flow (left) and in RMF-driven flow (right).

Performance tests have been conducted on a SGI Origin 3800 system at TU Dresden. Three cases with grids ranging from 110,000 up to 7.1 million elements were considered. In the tests, one time step with a fixed number of iterations

was performed using one to 120 processors. The measured speedup and the computational rate are depicted in Fig. 2. Especially for the medium grid a significant superlinear speedup is observed. Since the parallel efficiency is unlikely to increase, this behavior can only be attributed to gains in local efficiency. A possible explanation is that our code heavily depends on indirect addressing, which on its part results in a suboptimal cache efficiency. With decreasing size the problem fits better into the cache hierarchy, leading to a gradual improvement of local computational efficiency. We remark that similar effects were observed by other authors [9].

For the other two cases the same arguments apply. The larger problem does not fit in cache in 64 or less processors but shows a superlinear performance gain for higher numbers. Finally, the smallest problem achieves optimal cache performance earlier, and thus shows an increasing loss of parallel efficiency. Therefore, the overhead introduced for such a small problem is obvious. However, when taking into account that the local grids contain only about one thousand elements (or 200 nodes) on 120 processors, the measured efficiency of 75% is still remarkable.

While the parallel speedup is dependent on the total problem size, the computational rate (average local size divided by wall clock time) shows a consistent behavior: Unless the local problem does not fit in cache a sustained rate of 2,500 elements per second is obtained in all three cases. The highest rate and, hence, optimal cache performance is achieved with 10 thousand (or less) local elements. From this point, decreasing parallel efficiency starts to reduce the computational rate.

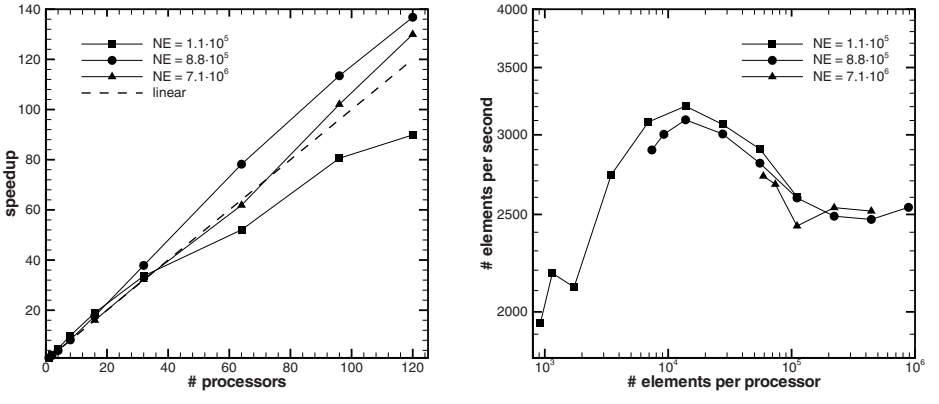

Fig. 2. Speedup (left) and computational rate (right).

5 Application

The solver is currently being used to perform direct numerical simulations of the flow in a cylindrical cavity that is exposed to a rotating magnetic field (RMF). This problem is related to electromagnetic stirring of melts in metallurgy and crystal growth in semiconductor production. Despite of its significance, the transitional and early turbulent regimes of flow are largely unexplored.

Figure 3 depicts the configuration. $\boldsymbol{B}$ is the flux density vector of the magnetic field which rotates with angular speed ω. Comparable to an induction motor, the field induces a primary rotating motion of the enclosed electro-conducting fluid. Additionally, a secondary recirculating flow develops due to the local action of friction forces. Under the so-called low-frequency/low-induction assumption, which is valid for most practical applications, the flow is described by the incompressible Navier-Stokes equations with an a priori known Lorentz force (see [10] for more details).

In our simulations computational grids ranging from 10^5 up to 20 million elements (3.5 million nodes) have been applied. Figure 3 shows a typical coarse grid divided into 32 partitions. An important result of the DNS study is the insight in the formation, evolution and finally dissipation of Taylor-Görtler-like vortices that obviously dominate the turbulence physics and provide an efficient mixing mechanism in this type of flow (Fig. 4). A detailed description of this and further results will be subject to a forthcoming paper.

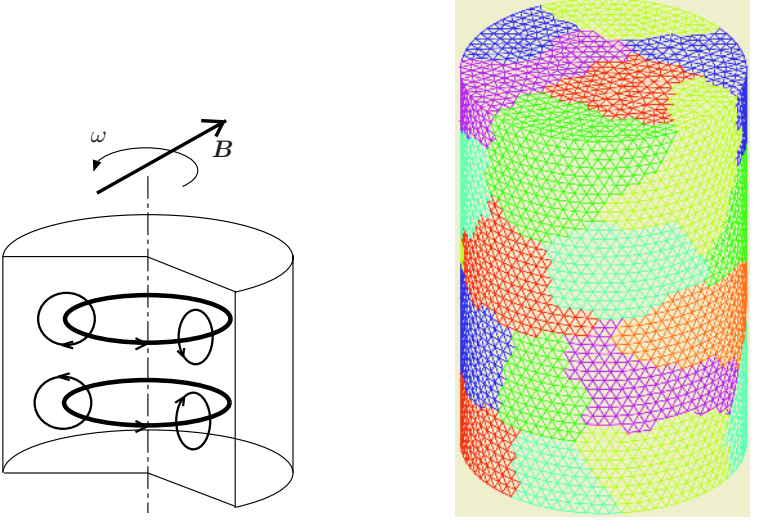

Fig. 3. Stirring with a rotating magnetic field: Sketch of configuration (left) and coarse grid divided into 32 partitions (right).

Fig. 4. Instantaneous vortex structures in weakly turbulent regime.

6 Conclusion

We have described a splitting approach to the pressure-stabilized Petrov-Galerkin finite element method for incompressible flow. Similar to conventional fractional step or projection methods it allows for segregating the pressure computation but retains a consistent formulation. The existing MG grid library provided a suitable basis for the parallel implementation of the method. The excellent scalability of the flow solver was demonstrated in a performance study on a SGI Origin 3800 using up to 120 processors. These tests also revealed a considerable superlinear speedup which can be explained by increasing cache efficiency at smaller local problem sizes. The solver was successfully applied in large production runs for direct numerical simulations of transitional and turbulent flows driven by rotating magnetic fields.

Acknowledgments

Financial support from German “Deutsche Forschungsgemeinschaft” in frame of the Collobarative Research Center SFB 609 is gratefully acknowledged. The measurements were done on a SGI Origin 3800 based on a grant from ZHR at TU Dresden.

References

1. Hughes, T.J.R., Franca, L.P., Balestra, M.: A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška-Brezzi condition: A stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations. Comput. Meth. Appl. Mech. Eng. **59** (1986) 85–99.
2. Codina, R.: Pressure Stability in Fractional Step Finite Element Methods for Incompressible Flows. J. Comp. Phys. **170** (2001), 112–140.
3. Stiller, J., Nagel, W.E.: MG – A Toolbox for Parallel Grid Adaption and Implementing Unstructured Multigrid Solvers. In: E.H. D'Hollander et al. (Eds.): Parallel Computing. Fundamentals & Applications. Imperial College Press 2000, 391–399.
4. Tezduyar, T., Osawa, Y.: Finite element stabilization parameters computed from element matrices and vectors. Comput. Meth. Appl. Mech. Eng. **190** (2001), 411–430.
5. Gresho, P.M.: On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a nite element method that also introduces a nearly consistent mass matrix I: Theory. Int. J. Num. Meth. Fluids. **11** (1990), 587–620.
6. Stiller, J., Wienken, W., Fladrich, U., Grundmann, R., Nagel, W.E.: Parallel and Adaptive Finite Element Techniques for Flow Simulation. In C. Breitsamter et al. (Eds.): New Results in Numerical and Experimental Fluid Mechanics IV (Notes on Numerical Fluid Mechanics and Multidisciplinary Design, Vol. 87). Springer Verlag 2004, 366–373.
7. Karypis, G., Kumar, V.: Multilevel Algorithms for Multi-Constraint Graph Partitioning. Univ. Minnesota, Dep. Computer Science, TR 98-019, 1998.
8. Davidson, P.A.: An Introduction to Magnetohydrodynamics. Cambridge University Press 2001.
9. Baggag, A., Atkins, H., Keyes, D.E.: Parallel implementation of the Discontinous Galerkin Method. In: Proceedings of Parallel CFD'99, 1999, 233–240.
10. Gelfgat, Yu.M., Priede, J.: MHD flows on a rotating magnetic field (A review). Magnetohydrodynamics **31** (1995) 1–2, 188–200.