# ON SMOOTHERS IN PARALLEL COUPLED MULTIGRID METHODS FOR INCOMPRESSIBLE NAVIER–STOKES EQUATIONS

VOLKER JOHN *

**Abstract.** We present a numerical study of two types of smoothers in parallelized coupled multigrid methods for the solution of incompressible Navier–Stokes equations. The Vanka smoother is characterized by the solution of many small linear systems of equations in each smoothing step whereas the Braess–Sarazin smoother solves one large linear saddle point problem. Our study is based on the nonconforming $P_1/P_0$–finite element discretization in space and the Crank–Nicolson discretization in time. The computations were performed on parallel computer HP N–Class with RISC 8500 processors using the communication interface MPI. The Vanka smoother shows a better efficiency in this framework.

**Key words.** incompressible Navier–Stokes equations, coupled multigrid methods, Vanka smoother, Braess–Sarazin smoother

**AMS subject classifications.** 65M55, 65N22, 65N55, 65Y05

**1. The incompressible Navier–Stokes equations and their finite element discretization.** We consider the incompressible Navier–Stokes equations

$$
\begin{array}{rcll}
\dfrac{\partial \mathbf{u}}{\partial t} - \nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p & = & \mathbf{f} & \text{in } \Omega\times(0,T], \\
\nabla\cdot\mathbf{u} & = & 0 & \text{in } \Omega\times(0,T], \\
\mathbf{u} & = & \mathbf{g} & \text{on } \partial\Omega_D\times(0,T], \\
-\nu\nabla\mathbf{u}\cdot\mathbf{n}_{\partial\Omega_O} + p\mathbf{n}_{\partial\Omega_O} & = & \mathbf{0} & \text{on } \partial\Omega_O\times(0,T], \\
\mathbf{u} & = & \mathbf{u}_0 & \text{in } \Omega \text{ for } t=0,
\end{array}
\tag{1.1}
$$

where $\Omega$ is a bounded domain in $\mathbb{R}^2$ with boundary $\partial\Omega = \partial\Omega_D \cup \partial\Omega_O$, $\partial\Omega_D \cap \partial\Omega_O = \emptyset$, $\mathbf{u}$ the velocity, $p$ the pressure, $\nu$ the kinematic viscosity of the fluid, $\mathbf{g}$ a Dirichlet boundary condition on the Dirichlet boundary $\partial\Omega_D$, $\partial\Omega_O$ the outflow boundary with the outward directing normal $\mathbf{n}_{\partial\Omega_O}$, $\mathbf{u}_0$ an initial velocity, $T$ the end of the time interval, and $\mathbf{f}$ represents exterior forces. Besides (1.1), we study also the steady state Navier–Stokes equations where $\partial\mathbf{u}/\partial t \equiv 0$.

As first step, (1.1) is discretized in time by the Crank–Nicolson scheme. The arising nonlinear equation in each time step is linearized then by a fixed point iteration. In each nonlinear iteration step, a so–called Oseen equation

$$
\begin{array}{rcll}
\dfrac{\mathbf{u}^{n+1}}{\Delta t} - \nu\Delta\mathbf{u}^{n+1} + (\mathbf{u}^n\cdot\nabla)\mathbf{u}^{n+1} + \nabla p^{n+1} & = & \bar{\mathbf{f}} & \text{in } \Omega, \\
\nabla\cdot\mathbf{u}^{n+1} & = & 0 & \text{in } \Omega, \\
\mathbf{u}^{n+1} & = & \mathbf{g} & \text{on } \partial\Omega_D, \\
-\nu\nabla\mathbf{u}^{n+1}\cdot\mathbf{n}_{\partial\Omega_O} + p^{n+1}\mathbf{n}_{\partial\Omega_O} & = & \mathbf{0} & \text{on } \partial\Omega_O,
\end{array}
\tag{1.2}
$$

with the given iterate $(\mathbf{u}^n, p^n)$ has to be solved. Here, $\Delta t$ denotes the length of the current time step and $\bar{\mathbf{f}}$ is the sum of $\mathbf{f}$ and terms arising in the temporal discretization of (1.1). Problem (1.2) is discretized in space by the nonconforming $P_1/P_0$–finite element discretization from Crouzeix and Raviart [4]. To apply this discretization, $\Omega$ is decomposed into triangles and, if necessary, the boundary $\partial\Omega$ is approximated by

*Institut für Analysis und Numerik, Otto–von–Guericke–Universität Magdeburg,Postfach 4120, 39016 Magdeburg, Germany, phone +49 391 67 12633, fax +49 391 67 18073, john@mathematik.uni-magdeburg.de, http://david.math.uni-magdeburg.de/~home/john/

a polygon with the vertices of the polygon on $\partial\Omega$. The discrete velocity is computed in the finite element space

$$V_h := \left\{ \begin{array}{l} \text{space of vector–valued piecewise linear functions which are} \\ \text{continuous at the midpoints of edges of the triangulation} \end{array} \right\}$$

whereas the discrete pressure belongs to the space

$$Q_h := \Big\{ \text{space of piecewise constant functions} \Big\}.$$

If $\partial\Omega_O = \emptyset$, it is required in addition that the discrete pressure has zero mean value, i.e. $\int_\Omega p_h dx = 0$. Some advantages of this discretization for the numerical solution of flow problems are discussed in Section 2.5.

An additional stabilization is necessary for convection dominated flows. We use the Samarskij upwinding stabilization analyzed by Schieweck and Tobiska [14] for the nonconforming $P_1/P_0$–finite element discretization of the steady state Navier–Stokes equations.

Altogether, an iterative solution of the linearized and discretized incompressible Navier–Stokes equations requires over and over again the approximate solution of large and sparse saddle point problems of the form

$$\mathcal{A} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}. \tag{1.3}$$

**2. The coupled multigrid solver.** Coupled multigrid methods compute the solution for both types of unknowns (velocity and pressure) of (1.3) simultaneously. Benchmark computations (Schäfer and Turek [13]) show that these methods are one of the best class of solvers known so far for laminar Navier–Stokes equations.

The efficient numerical solution of complex problems in fluid mechanics is often only possible on parallel computers. However, the parallel efficiency of multigrid methods is not always convincing. Often, the multigrid V–cycle shows a good parallel but a not satisfying numerical efficiency whereas the situation in the W–cycle is vice versa. Numerical studies, e.g. in [10], show that the F–cycle is a good compromise on parallel computers. For this reason, we will present only results for the F–cycle.

**2.1. The Vanka smoother.** Vanka–type (Vanka [16], John [6, 7]), John and Tobiska [10], or local (Turek [15]), or coupled or segregated (Paisley and Bhatti [11]) smoothers, can be considered as block Gauss–Seidel methods where one block consists of a small number of degrees of freedom (d.o.f.). The characteristic feature of this type of smoother is that in each smoothing step a large number of small linear systems of equations has to be solved.

In the (cell wise) Vanka smoother, a block corresponds to all degrees of freedom which are connected to one mesh cell $K$. For the nonconforming $P_1/P_0$–finite element discretization, these are six velocity degrees of freedom (two at the midpoint of each edge) and one pressure degree of freedom. Thus, a smoothing step with the Vanka smoother consists in a loop over all triangles where in each triangle a $7\times 7$ linear system of equations has to be solved. The degrees of freedom are updated in a Gauss–Seidel manner. Since each velocity degree of freedom in the interior of $\Omega$ belongs to two triangles, it is updated twice in one smoothing step.

We denote by $\mathcal{A}_K$ the block of the matrix $\mathcal{A}$ which is connected to the degrees of freedom of the triangle $K$, i.e. the intersection of the rows and columns of $\mathcal{A}$ which are associated with $K$. We define

$$\mathcal{A}_K = \begin{pmatrix} A_K & B_K \\ B_K^T & 0 \end{pmatrix} \in \mathbb{R}^{7\times 7} \text{ and } \mathcal{D}_K = \begin{pmatrix} \operatorname{diag}(A_K) & B_K \\ B_K^T & 0 \end{pmatrix} \in \mathbb{R}^{7\times 7}.$$

Similarly, we denote by $(\cdot)_K$ the restriction of a vector on the rows corresponding to the degrees of freedom connected to $K$. The (diagonal) Vanka smoother updates the velocity and pressure values in the triangle $K$ by

$$\begin{pmatrix} u \\ p \end{pmatrix}_K := \begin{pmatrix} u \\ p \end{pmatrix}_K + \mathcal{D}_K^{-1}\left(\begin{pmatrix} f \\ g \end{pmatrix} - \mathcal{A}\begin{pmatrix} u \\ p \end{pmatrix}\right)_K.$$

In order to avoid communications within a smoothing iteration, we apply the Vanka smoother in parallel on each processor. After each smoothing iteration, the values at the interfaces are averaged by one communication step. The numbering of the triangles within the loop over the triangles influences its behaviour. In the tests presented below, the triangles are ordered lexicographically: a triangle with barycentre coordinates $(x_0, y_0)$ is predecessor of a triangle on the same processor with barycentre coordinates $(x_1, y_1)$ iff $(x_0 < x_1) \vee ((x_0 = x_1) \wedge (y_0 < y_1))$.

**2.2. The Braess–Sarazin smoother.** The Braess–Sarazin smoother, introduced by Braess and Sarazin in [2], solves in each smoothing step one large linear system of equations which is simpler to solver than (1.3). This type of smoother is called in the literature also global, decoupled or segregated smoother.

Let $u^k$ be given. Then, the Braess–Sarazin smoother solves in one smoothing step the global linear saddle point problem

$$\begin{pmatrix} \alpha C & B \\ B^T & 0 \end{pmatrix}\begin{pmatrix} u^{k+1} \\ p^{k+1} \end{pmatrix} = \begin{pmatrix} f - (A - \alpha C)u^k \\ g \end{pmatrix}, \tag{2.1}$$

where $(\alpha C)^{-1}$ is an approximation to $A^{-1}$. The matrix $C$ should be easier invertible than $A$. Solving the upper equation for $u^{k+1}$ and inserting the result into the lower equation lead to the pressure Schur complement system

$$(B^T C^{-1} B)p^{k+1} = -(B^T C^{-1}(Au^k - f) + \alpha(g - B^T u^k)). \tag{2.2}$$

The efficient solution of (2.2) is the core of an efficient solution of (2.1). In the application of the Braess–Sarazin smoother, system (2.2) needs to be solved only approximately. After having computed $p^{k+1}$ from (2.2), $u^{k+1}$ can be obtained from the upper equation of system (2.1).

The Braess-Sarazin smoother can be considered as a pressure Schur complement scheme. The most famous representative of this class of schemes is SIMPLE by Patankar and Spalding [12]. SIMPLE is also often used as smoother in coupled multigrid methods, see e.g. [5]. However, both types of smoothers differ in important properties, e.g. the error in the Braess–Sarazin smoother depends only on the old velocity $u^k$ but not on the old pressure $p^k$ like in SIMPLE. The convergence of the coupled multigrid W–cycle with the Braess–Sarazin smoother can be proven for the Stokes equations, $C = I$ (identity), and inf–sup stable conforming and nonconforming finite element discretizations if $\alpha$ is chosen large enough, [2, 9]. It was observed in [10] also for the Navier–Stokes equations that $\alpha$ has to be sufficiently large. There is an example in [2] which shows that SIMPLE does not possess always good smoothing properties.

An important point is the choice of the approximation $(\alpha C)^{-1}$ of $A^{-1}$. In the steady state Oseen equations, $A$ is in general nonsymmetric. We used in the numerical tests presented below $(\alpha C)^{-1} = \alpha^{-1}[ILU_1(0)(A)]^{-1}$, i.e. a block ILU–decomposition of $A$ without additional fill-in and the absolute value of a dropped entry is added to the main diagonal of the corresponding row. A block corresponds to all velocity degrees of freedom which are stored on one processor and within the blocks a lexicographical ordering is used. The arising Schur complement matrix $B^T[ILU_1(0)(A)]^{-1}B$ cannot be stored explicitely in an efficient way and system (2.2) is solved approximately with 10 steps of GMRES. Since the system matrix in (2.2) is only known implicitly, we could not construct an appropriate preconditioner for GMRES. Another drawback of this approach are the additional memory requirements of GMRES. We have used the damping parameter $\alpha = 1.25$.

The situation for the time dependent Oseen equations is different and more favourable. There, the matrix $A$ is, at least for small time steps, diagonally dominant. Thus, $(\alpha C^{-1}) = \alpha^{-1}\mathrm{diag}(A)^{-1}$ is a good approximation. In addition, the nonconforming $P_1/P_0$–finite element discretization offers an easy way to store the Schur complement matrix $B^T(\mathrm{diag}(A))^{-1}B$ explicitly. This matrix is symmetric and positive definite. For the approximate solution of (2.2), we used 3 steps of a preconditioned conjugate gradient method with diagonal preconditioner. The damping parameter is set to be $\alpha = 1.25$.

The application of the Braess–Sarazin smoother requires the choice of more parameters than the use of the Vanka smoother. A good choice of this set of parameters might be difficult in practice and can be accomplished often only by numerical tests.

**2.3. Other components of the coupled multigrid method.** The grid transfer operations are usual defect restriction and function prolongation. The system on the coarsest grid is solved approximately with the Vanka smoother, i.e. the same coarse grid solver was used in all tests. We applied 100 iterations on the coarsest grid in the steady state problems and 10 iterations in the time dependent problems.

**2.4. Recent comparisons of smoothers in the literature.** Vanka and pressure Schur complement smoothers have been compared recently in at least two papers [11, 10]. The framework of these comparisons as well as the results of the numerical tests were different. In [10], the same framework as here was used and the Vanka smoother behaved more efficient than the Braess–Sarazin smoother in all tests. In [11], the steady state Navier–Stokes equations were considered with an additional transport equation. The equations were discretized by a finite volume method on rectangular grids. The convective terms were dealt with a second order flux–limited scheme. SIMPLE is used as a pressure Schur complement smoother and numerical results for the W(1,1)–cycle on a workstation were presented. In both numerical studies, the Vanka, called SCGS in [11], behaved better for the lid driven cavity problem but SIMPLE was superior for other problems. The different results show that smoothers in coupled multigrid methods need still more investigations in order to achieve a general evidence about their behaviour and efficiency.

**2.5. Some properties of the nonconforming $P_1/P_0$–finite element discretization.** The nonconforming $P_1/P_0$–finite element discretization fulfills the inf–sup or Babuška–Brezzi stability condition uniformly with respect to the mesh size and the shape regularity constant of the mesh. This guarantees the unique solvability of the arising discrete systems. Besides this favourable analytical property, it has also advantages from the point of view of the implementation on a parallel computer. The degrees of freedom of the velocity are associated with the midpoints of the edges of the triangles. Thus, those degrees of freedom which belong to an interface between two subdomains, each of which stored on a separate processor, have to be stored on two processors only. For this reason, only one communication is needed to interchange information between the same degree of freedom on different processors. This is not true for conforming finite element spaces where the degrees of freedom are connected with the vertices of mesh cells. In this case, the degrees of freedom at so–called crosspoints belong to and have to be stored on more than two processors. As a consequence, a complete interchange of information in case of crosspoints requires more than one communication. This results in an increase of the communication overhead.

A drawback of this discretization is its low accuracy. A recent study on a benchmark problem for the steady state Navier–Stokes equations, John and Matthies [8], shows that results of the same accuracy could be obtained with higher order finite element discretizations with much less degrees of freedom. But a difficulty of higher order discretizations was the solution of the discrete systems. In [8] it was found that a modified multigrid solver with the higher order discretization on the top of the multigrid (level $L+1$) and the remaining multigrid hierarchy built by a first order nonconforming upwind discretization could overcome this difficulty in many situations, see Figure 2.1. One core of the success of such a multigrid solver is a highly efficient multigrid method for first order nonconforming upwind discretizations as it is studied here.

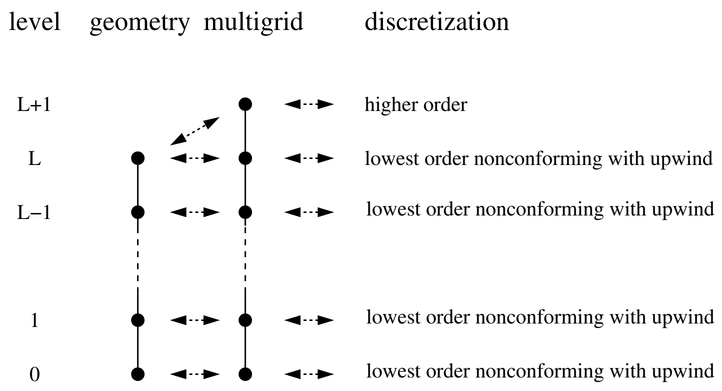


FIG. 2.1. *Modified multigrid method for higher order discretizations.*

**3. Numerical studies.** All computations were performed on a parallel computer HP N–Class with PA–RISC 8500 processors and shared memory. The communication among the processors was realized with the interface MPI.

The implementation of the algorithms was done using modules of the program package *ugp*1.0 which is an early version of the now available program *UG*3 from Bastian et al. [1]. In particular, data structures, load balancing routines and the parallel environment of *ugp*1.0 were used. However, the data structures had to be extended in order to handle multidimensional data, nonconforming finite elements, Navier–Stokes equations, and time dependent problems.

For the steady state equations, the initial guess for the fixed point iteration on the finest level was the interpolation of the solution on the previous level. The iteration was stopped for an Euclidean norm of the residual less than $10^{-10}$. In the time dependent problems, the solution of the previous time step was used as initial iterate for the nonlinear iteration and this iteration was stopped for an Euclidean norm of the residual less than $10^{-6}$. The linear saddle point problem (1.3) in each step of the fixed point iteration was solved up to a reduction of the Euclidean norm of the residual by the factor 10.

**3.1. Steady state channel flow past a cylinder.** We consider a benchmark problem defined in the DFG priority research program *Flow Simulation with High–Performance Computers*, [13]. This benchmark problem describes the steady state flow in a channel around a cylinder, see Figure 3.1 for the domain and the initial grid (level 0). The parameters of the Navier–Stokes equations (1.1) are $\nu = 10^{-3}$, $\mathbf{f} = 0$, the parabolic inflow profile $\mathbf{u}(0, y) = \mathbf{u}(2.2, y) = 0.41^{-2}(1.2y(0.41 - y), 0)$, $0 \le y \le 0.41$, outflow boundary conditions at the outlet, and no–slip conditions on the other boundaries. The resulting flow possesses two large vertices behind the cylinder. We present in Table 3.1 results on level 6 (4 763 684 velocity d.o.f. and 1 589 248 pressure d.o.f.) which were computed on 8 processors.

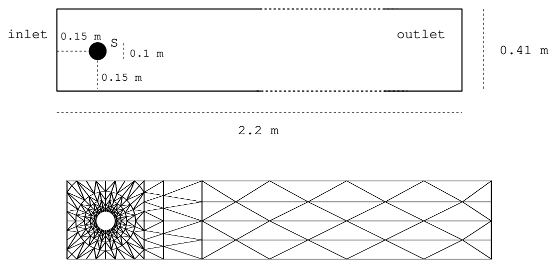


FIG. 3.1. *Domain and initial grid for the channel flow past a cylinder.*

TABLE 3.1
*Computing times for the channel flow past a cylinder.*

| | Vanka | | | | Braess–Sarazin | | | |
|---|---|---|---|---|---|---|---|---|
| cycle | F(2,2) | F(3,3) | F(4,4) | F(5,5) | F(2,2) | F(3,3) | F(4,4) | F(5,5) |
| time (s) | div. | 3822 | 2314 | 2306 | 11490 | 11800 | 13350 | 13970 |

The coupled multigrid method with the Vanka smoother computes the solution much faster than with the Braess–Sarazin smoother. The higher efficiency of the Vanka smoother does not come from its smoothing property. In contrast, the Braess–Sarazin smoother shows a better smoothing property since the F(2,2)–cycle with this smoother converges. The reason for the higher efficiency of the Vanka smoother is mainly the lower computational costs per smoothing step. In addition, the Vanka smoother benefits from the current hardware architecture since the solution of its small systems of equations can be done very fast in the cache memory.

**3.2. High Reynolds number lid driven cavity flow.** We study the steady state lid driven cavity problem for the Reynolds number $Re = \nu^{-1} = 3200$. This problem is defined in the unit square. The right hand side $\mathbf{f}$ is zero. No–slip boundary conditions are imposed for $x = 0, x = 1$, and $y = 0$. The velocity at the upper boundary is given by $\mathbf{g} = (1, 0)$. The coarsest grid has 32 triangles and it originates form the grid shown in Figure 3.2 by one red refinement step. The computations with 1 570 816 velocity d.o.f. and 524 288 pressure d.o.f. were performed on 4 processors.

TABLE 3.2
*Computing times and communication overhead for the lid driven cavity problem, $Re = 3200$.*

| | Vanka | | | Braess–Sarazin | | |
|---|---|---|---|---|---|---|
| cycle | F(2,2) | F(3,3) | F(4,4) | F(2,2) | F(3,3) | F(4,4) |
| time (s) | 6415 | 6678 | 7877 | 31300 | 41270 | 52460 |
| communication (%) | 4.4 | 3.3 | 3.4 | 7.5 | 9.0 | 7.5 |

Table 3.2 shows the higher efficiency of the Vanka smoother for the high Reynolds number problem. The communication overhead of this smoother is also less than for the Braess–Sarazin smoother. The results presented in Table 3.2 are somewhat different to results on the parallel computer Parsytec GCPowerPlus [10]. On this older parallel computer, the differences in the communication overhead were much larger but the differences in the computing times considerably smaller. We think that the improvement of the computation speed within the cache memory in the last couple of years is one reason for the differences in the results. This very fast computation speed is exploited by the Vanka smoother but not by the Braess–Sarazin smoother.

**3.3. The time dependent vortex decay problem.** Chorin's vortex decay problem [3] is defined in the unit square and the solution of (1.1) with $\mathbf{f} = \mathbf{0}$ is

$$\begin{aligned} \mathbf{u} &= \exp(-2n^2\pi^2\nu t)\begin{pmatrix} -\cos(n\pi x)\sin(n\pi y) \\ \sin(n\pi x)\cos(n\pi y) \end{pmatrix} \\ p &= -0.25\exp(-4n^2\pi^2\nu t)\left(\cos(2n\pi x) + \cos(2n\pi y)\right). \end{aligned}$$

We applied Dirichlet boundary conditions and performed computations for $n = 4$, $\nu = 10^{-4}$, $T = 10$, $\Delta t = 0.01$, see Figure 3.2 for the streamlines of the solution and the coarsest grid. We present in Table 3.3 computations on 4 processors ( 392 192 velocity d.o.f., 131 072 pressure d.o.f.) and on 8 processors ( 1 570 816 velocity d.o.f., 524 288 pressure d.o.f.).

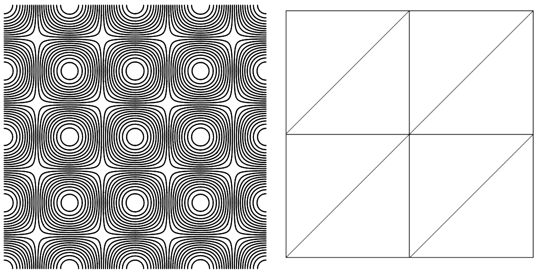

FIG. 3.2. *Streamlines and initial triangulation for Chorin's vortex decay problem.*

The Vanka smoother worked also for the time dependent problem more efficient, see Table 3.3. But its superiority is less than in the steady state case. The explicit storage of the Schur complement matrix $B^T(\mathrm{diag}(A))^{-1}B$ in the Braess–Sarazin smoother and the preconditioner in the conjugate gradient method enhance the efficiency of this smoother considerably.

**3.4. Summary of the numerical tests.** The Vanka smoother worked more efficient than the Braess–Sarazin smoother in the presented framework. This higher

TABLE 3.3
*Computing times and communication overhead for Chorin's vortex decay problem, above 4 processors, below 8 processors.*

| | Vanka | | | Braess–Sarazin | | |
|---|---|---|---|---|---|---|
| cycle | F(1,1) | F(2,2) | F(3,3) | F(1,1) | F(2,2) | F(3,3) |
| time (s) | div. | 34020 | 43120 | 96148 | 52905 | 68030 |
| communication % | | 0.7 | 1.1 | 1.6 | 0.9 | 2.2 |
| time (s) | div. | 102018 | 104005 | 230502 | 146413 | 166602 |
| communication % | | 3.1 | 3.1 | 3.5 | 3.1 | 2.7 |

efficiency can be clearly seen for the steady state Navier–Stokes equations and it is less dramatically for the time dependent Navier–Stokes equations. The superiority of the Vanka smoother is mainly caused by the lower computational costs per smoothing step. In addition, this smoother has a lower communication overhead and it takes advantage of very fast cache memory. The Braess–Sarazin smoother shows a better smoothing property but its computational costs are simply too large.

## REFERENCES

[1] P. Bastian, K. Birken, K. Johannsen, S. Lang, N. Neuß, H. Rentz-Reichert, and C. Wieners. UG - a flexible software toolbox for solving partial differential equations. *Comput. Visual. Sci.*, 1(1):27 – 40, 1997.
[2] D. Braess and R. Sarazin. An efficient smoother for the Stokes problem. *Applied Numer. Math.*, 23(1):3 – 19, 1997.
[3] A.J. Chorin. Numerical solution of the Navier–Stokes equations. *Math. Comp.*, 22:745 – 762, 1968.
[4] M. Crouzeix and P.-A. Raviart. Conforming and nonconforming finite element methods for solving the stationary Stokes equations I. *R.A.I.R.O. Analyse Numérique*, 7:33–76, 1973.
[5] T. Gjesdal and M.E.H. Lossius. Comparison of pressure correction smoothers for multigrid solutions of incompressible flow. *Int. J. Num. Meth. Fluids*, 25:393 – 405, 1997.
[6] V. John. On the parallel performance of coupled multigrid methods for the solution of incompressible Navier–Stokes equations. In M. Griebel et al., editor, *Large–Scale Scientific Computations of Engineering and Environmental Problems*, volume 62 of *Notes on Numerical Fluid Mechanics*, pages 269 – 280. Vieweg, 1998.
[7] V. John. A comparison of parallel solvers for the incompressible Navier–Stokes equations. *Comput. Visual. Sci.*, 1(4):193 – 200, 1999.
[8] V. John and G. Matthies. Higher order finite element discretizations in a benchmark problem for incompressible flows. submitted, 2000.
[9] V. John and L. Tobiska. A coupled multigrid method for nonconforming finite element discretizations of the 2d–Stokes equation. *Computing*, 2000. to appear.
[10] V. John and L. Tobiska. Smoothers in coupled multigrid methods for the parallel solution of the incompressible Navier–Stokes equations. *Int. J. Num. Meth. Fluids*, 2000. to appear.
[11] M.F. Paisley and N.M. Bhatti. Comparison of multigrid methods for neutral and stably stratified flows over two–dimensional obstacles. *J. Comput. Phys.*, 142:581 – 610, 1998.
[12] S.V. Patankar and D.B. Spalding. A calculation procedure for heat and mass transfer in three–dimensional parabolic flows. *Int. J. Heat Mass Transfer*, 15:1787 – 1806, 1972.
[13] M. Schäfer and S. Turek. The benchmark problem "Flow around a cylinder". In E.H. Hirschel, editor, *Flow Simulation with High–Performance Computers II*, volume 52 of *Notes on Numerical Fluid Mechanics*, pages 547 – 566. Vieweg, 1996.
[14] F. Schieweck and L. Tobiska. An optimal order error estimate for an upwind discretization of the Navier–Stokes equations. *Num. Meth. Part. Diff. Equ.*, 12:407 – 421, 1996.
[15] S. Turek. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach*, volume 6 of *Lecture Notes in Computational Science and Engineering*. Springer, 1999.
[16] S. Vanka. Block-implicit multigrid calculation of two-dimensional recirculating flows. *Comp. Meth. Appl. Mech. Eng.*, 59(1):29–48, 1986.