Performance of the HYMLS multilevel ILU preconditioner on 3D flow equations

Sven Baars[a], Mark van der Klok[a], Weiyan Song[a,b], Jonas Thies[b], Arthur Veldman[a], Fred Wubs[a]

[a] *Univ. of Groningen, Bernoulli Inst. for Mathematics, Computer Science and Artificial Intelligence*
[b] *German Aerospace Center, Inst. Simulation and Software Technology*

Abstract

This paper discusses the solution of the three dimensional, steady, incompressible Navier-Stokes equations using a fully coupled iterative approach. We discuss and demonstrate the advantages of avoiding time integration, and of the fully coupled incomplete LU (ILU) factorization (as opposed to block preconditioners trying to split the problem into scalar subsystems). The paper introduces the idea of skew-partitioning to overcome practical issues encountered in the recursive computation of the multi-level ILU on staggered grids.

Keywords: 3D incompressible Navier-Stokes, Arakawa C-grid, continuation method, fully coupled solver, multi-level incomplete factorization
2000 MSC: 65F08, 65F50, 65Y05, 68W10, 76E09

1. Introduction

In this paper, we are concerned with the solution of the linear systems that occur after linearization and (implicit) discretization of the incompressible Navier-Stokes equations describing the motion of viscous, incompressible fluids. In contrast to common practice, we employ Newton's method for linearizing the equations and solve for steady states directly (without a time stepping scheme). These aspects make the overall approach very efficient, but the solution of the linear systems becomes challenging and calls for dedicated research on the sparse linear solver.

In dimensionless conservative form, the equations are given by

$$\frac{\partial \mathbf{u}}{\partial t} + (\nabla \cdot (\mathbf{u} \cdot \mathbf{u}^T))^T = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \mathbf{u}, \tag{1}$$

$$0 = \nabla \cdot \mathbf{u}. \tag{2}$$

Here Re is the Reynolds Number defined as $\mathrm{Re} = \frac{L\rho U}{\eta}$ with U and L the characteristic velocity and length scales, respectively, and ρ and η the density and dynamic viscosity, respectively. The system is closed by imposing appropriate boundary and initial conditions.

For the discretization of these equations the finite volume method on a staggered C-grid is employed, where the unknowns are positioned as indicated in Fig. 1. This discretization is quite common for incompressible fluids. The differences and averages needed are always taken of the two nearest unknowns of the same type, leading to central second-order discretizations on smoothly varying grids.

The discretization leads to a system of ordinary differential equations (ODEs)

$$M \frac{d\mathbf{u}}{dt} = -N(\mathbf{u}, \mathbf{u}) + \frac{1}{Re} L\mathbf{u} - Gp + f, \tag{3}$$

$$0 = G^T \mathbf{u}. \tag{4}$$

Now $\mathbf{u}$ and p denote vectors representing the velocity and pressure in each grid point. $N(\cdot, \cdot)$ is a bilinear form arising from the convective terms, G is the discretization of the gradient operator; its transpose gives us the discretization of

Email address: s.baars@rug.nl (Sven Baars)

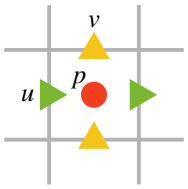


Figure 1: Positioning of variables on a grid cell in the 2D case.

the divergence operator, M is the mass matrix, containing the volumes of the grid cells on its diagonal and f contains the known part of the boundary conditions. For this exposition, we assume that we are computing the flow in a closed box, so all normal velocities at the walls are zero and hence the second equation, enforcing mass conservation, is homogeneous.

If we freeze one of the variables in a bilinear form then it becomes linear, hence one can write

$$N(\mathbf{u}, \hat{\mathbf{u}}) = N_1(\mathbf{u})\hat{\mathbf{u}} + N_2(\hat{\mathbf{u}})\mathbf{u}, \tag{5}$$

Using this notation, the system with the Jacobian, which typically has to be solved in a Newton step, is of the form

$$\begin{pmatrix} \frac{1}{\Delta t}M + N_1(\mathbf{u}) + N_2(\mathbf{u}) - \frac{1}{\mathrm{Re}}L & -G \\ D & O \end{pmatrix} \begin{pmatrix} \Delta\mathbf{u} \\ \Delta p \end{pmatrix} = -\begin{pmatrix} f_{\mathbf{u}} \\ f_p \end{pmatrix}, \tag{6}$$

where Δt is the time step used in an implicit time integrator, e.g. in backward Euler. If we assume that $N_1(\mathbf{u})$ gives just the discretized convection terms, then $N_2(\mathbf{u})$ will give the cross terms. It is known [31], that, on closed domains, dissipation of kinetic energy only occurs by diffusion. Our discretization preserves this property, which has the consequence that for divergence-free $\mathbf{u}$ the operator $N_1(\mathbf{u})$ is skew-symmetric. This means that if we omit $N_2(\mathbf{u})$, the approximate Jacobian will become negative definite on the space of discrete divergence-free velocities. The associated equations are called Oseen equations. This is advantageous for iterative procedures and the choice is therefore often made, e.g., [12]. In Section 3.4, we will see that this approach may fail for relatively low values of the Reynolds Number when computing steady states directly. Hence, we prefer to work with the full Jacobian.

We want to solve equation (6) in order to be able to study flow transitions. A flow transition is a qualitative change in the flow pattern when a physical parameter is changed only slightly. One example is the whistling sound which car mirrors used to make at high driving speeds. At low speeds the flow around the mirror will be steady, while at high speeds it will be oscillatory and produce sound waves. So qualitatively the flow changes from steady to oscillatory. This is one of the many practical examples one can give, and it is clear that industrial applications of numerical simulation of these equations can be found in many fields, such as in complex turbulent flows [9], aircraft design [32] and so on.

Mathematically a transition occurs as a bifurcation in a diagram depicting some typical flow quantity as a function of the parameter. In such plots one may observe different bifurcations, for all of them, eigenvalues of the system are important indicators, e.g. at an Hopf-bifurcation a conjugate pair of complex eigenvalues cross the imaginary axis and the imaginary part is predicting the frequency of the periodic solution which is emanating at that parameter value. Hence, determination of critical eigenvalues is important.

The obvious way to study transitions of flows is by just performing time integration for various values of natural parameters, e.g., the Reynolds Number. If a steady state is reached, then it is a stable steady state. If a periodic solution is reached, one has found a stable periodic solution. If the latter occurs after an increase of the natural parameter, then one knows that a Hopf-bifurcation has occurred during this increase. However, using the time integration in this way, only stable solutions can be found, though by a clever wrap around algorithm one can also deal with unstable cases [26, 18, 29].

Compared to time-marching approaches, the pseudo-arclength continuation method [14, 3] avoids potentially long time integrations to obtain generic and meaningful information and hence can bring down the computation time considerably, especially when calculating steady solutions [5]. In each continuation step, a nonlinear system has to be solved. For the solution of the nonlinear system, we have to resort to iterative methods to obtain an approximate solution in a finite number of iterations. The most frequently used methods are the Newton-Raphson method (Newton's

method for short) and the Picard iteration. Both lead to a linear system to be solved in each iteration and need a sufficiently accurate initial guess to achieve convergence.

Apart from continuation of solutions, we want to compute the critical eigenvalues to identify bifurcations under variation of parameters. This requires computing the most unstable eigenmode along with the steady-state flow, which can be more difficult than the computation of the flow itself. For large and non-symmetric systems, methods of choice are the Arnoldi method (with spectral transformation to target eigenvalues near 0), and the Jacobi-Davidson QR (JDQR) method [24, 27], which can elegantly exploit preconditioning techniques for linear systems. For completeness we mention that Meerbergen [20, 19] constructed an approach in which the parameter for which a Hopf bifurcation occurs can be computed directly by solving a Lyapunov equation.

The effectiveness of continuation and stability studies depends strongly on the efficient and robust solution of linear systems. The system 6 above is of saddle-point type. In [1], the authors gave a survey of methods currently used to solve linear systems of this type. Direct (sparse) solvers are not practical in 3D since a typical CFD problem involves millions of unknowns leading to a huge memory requirement for storing the factorization and an enormous amount of time to compute it. For such problems, iterative methods are preferred, e.g. Krylov subspace methods with suitable preconditioning to ensure robustness, fast convergence and accuracy of the final approximate solution [1, 2, 21, 7, 13, 6]. Segal et al. [25] give an overview of the present state of fast solvers for the incompressible Navier-Stokes equations discretized by the finite element method and linearized by Newton's or Picard's method. Preconditioners that are often advocated include standard additive-Schwarz domain-decomposition, multigrid with aggressive coarsening and strong smoothers (e.g. ILU), and 'block preconditioners' that use an approximate block LU factorization and some approximation of the Schur complement associated with the pressure unknowns, e.g. SIMPLEC, LSC, and PCD. The latter methods are implemented in the Trilinos package Teko [4] which we will use for comparison. This approach allows to apply highly optimized methods like multigrid (geometric or algebraic) on the separate blocks.

Another class of Schur complement methods is obtained when eliminating the interior of geometric partitions (or subdomains) and constructing a suitable approximation of the reduced system on the separator. In [33, 28] we showed how to construct a preconditioner of this kind for the C-grid discretization shown in Fig. 1. The resulting operator acts in the divergence free space, which allows the method to handle the saddle-point structure of the system in a natural way.

The algorithm and software are called HYMLS (a HYbrid, Multi-Level Solver) and the implementation uses the data structures available in the Trilinos software package Epetra and is therefore suitable for distributed memory computers. In this paper, we present the basic ideas of the multilevel variant of HYMLS and show its robustness on an academic problem, i.e., the lid-driven cavity problem, for various Reynolds Numbers. Though being academic, the problem is hard for preconditioners when the Reynolds Number is high, and hence it yields a discriminating test problem.

2. The HYMLS preconditioner

In [22] a direct method was proposed which is applicable to the equations and (C-grid) discretization studied here. It reduces fill and computation time while preserving the structure of the equations during the elimination. A two-level incomplete factorization method based on this approach was presented in [33]. It has the advantage that the ordering it defines for the matrix exposes parallelism on both levels: the computational domain is partitioned into a set of non-overlapping subdomains (the interiors) plus an interface (the separators). The Schur-complement problem on the separators is solved using a parallel incomplete factorization, and the systems associated with the subdomains can be solved independently using sequential sparse direct solvers.

2.1. Level 1: domain decomposition

The general idea can be illustrated as follows. For a general large (sparse) problem $K\vec{x} = \vec{b}$, after partitioning it into n subdomains and reordering the unknowns the problem can be written in the form

$$\begin{pmatrix} K_{11} & & & K_{1S} \\ & \ddots & & \vdots \\ & & K_{nn} & K_{nS} \\ K_{1S}^T & \dots & K_{nS}^T & K_{SS} \end{pmatrix} \begin{pmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_n \\ \vec{x}_S \end{pmatrix} = \begin{pmatrix} \vec{b}_1 \\ \vdots \\ \vec{b}_n \\ \vec{b}_S \end{pmatrix}.$$

The diagonal blocks K_{ii} (for $i = 1, \dots, n$) represent the interiors of the n subdomains (i.e. variables $\vec{x}_i$) and the blocks K_{iS} correspond to the couplings between these interior variables and the separator variables x_S. For ease of writing, we combine the block diagonal matrix containing all the interior blocks into a single block K_{II} and their couplings to the separators into K_{IS}, then, elimination of the interior variables formally yields the block LU decomposition

$$\begin{pmatrix} K_{II} & K_{IS} \\ K_{IS}^T & K_{SS} \end{pmatrix} = \begin{pmatrix} I & O \\ K_{IS}^T K_{II}^{-1} & I \end{pmatrix} \begin{pmatrix} K_{II} & K_{IS} \\ O & S \end{pmatrix}, \tag{7}$$

where

$$S = K_{SS} - K_{IS}^T K_{II}^{-1} K_{IS} = K_{SS} - \sum_{i=1}^{n} K_{iS}^T K_{ii}^{-1} K_{iS}$$

is the Schur-complement of K_{II}. The 'inverses' in the notation are evaluated using sparse LU factorizations. From the LU decomposition (Eq.7) we see that the partitioned system can now be solved by first solving

$$S\, x_S = b_S - \sum_{i=1}^{n} K_{iS}^T K_{ii}^{-1} b_i$$

followed by solving n independent systems

$$K_{ii} x_i = b_i - K_{iS}\, x_S, \qquad \forall i = 1, \dots, n,$$

which can be done in parallel.

2.2. Level 2: the approximate Schur-complement

The novel part in HYMLS is the way we construct a preconditioner for solving the unknowns on the separators occurring in the Schur-complement system. It represents the velocities surrounding the subdomains, and a single pressure per subdomain (a single pressure value is sufficient here to make the matrix non-singular). The velocities on the separators are grouped by the subdomains they belong to and the type of the variable (i.e. u, v, w), see also Figure 2.

The preconditioner is constructed by applying an orthogonal (Householder) transformation from the left and right to a block row and block column corresponding to each separator group in the Schur-complement. This transforms the unknowns of each separator group and we call the first one a V_Σ unknown and the remaining ones non-V_Σ unknowns, since it can be shown that the first one is an average of the original unknowns. Next, dropping all connections between non-V_Σ unknowns and V_Σ unknowns, and between non-V_Σ unknowns in different separator groups yields a block-diagonal preconditioner with dense blocks for the non-V_Σ, and a large and sparse block for the V_Σ unknowns. The transformation preserves the spectral properties of the original Schur-complement, whereas the dropping step reduces the fill of the Schur-complement matrix without harming the approximation properties substantially.

The 'V_Σ' part of the preconditioner (in which no dropping has occurred) turns out to have the same saddle point structure and numerical properties as the original matrix. It represents the relation between 'coarse grained' flow variables describing the pressure in each subdomain, and the flux through the interfaces between the subdomains. If we at this point use a sparse direct solver to factor this matrix, we obtain the two-level method from [33].

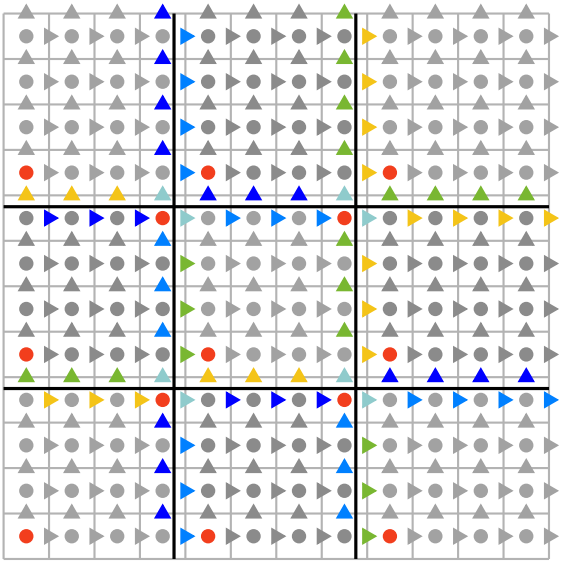


Figure 2: Construction of the preconditioner from the grid point-of-view. Every separator group has its own color.

2.3. Interplay of discretization and partitioning

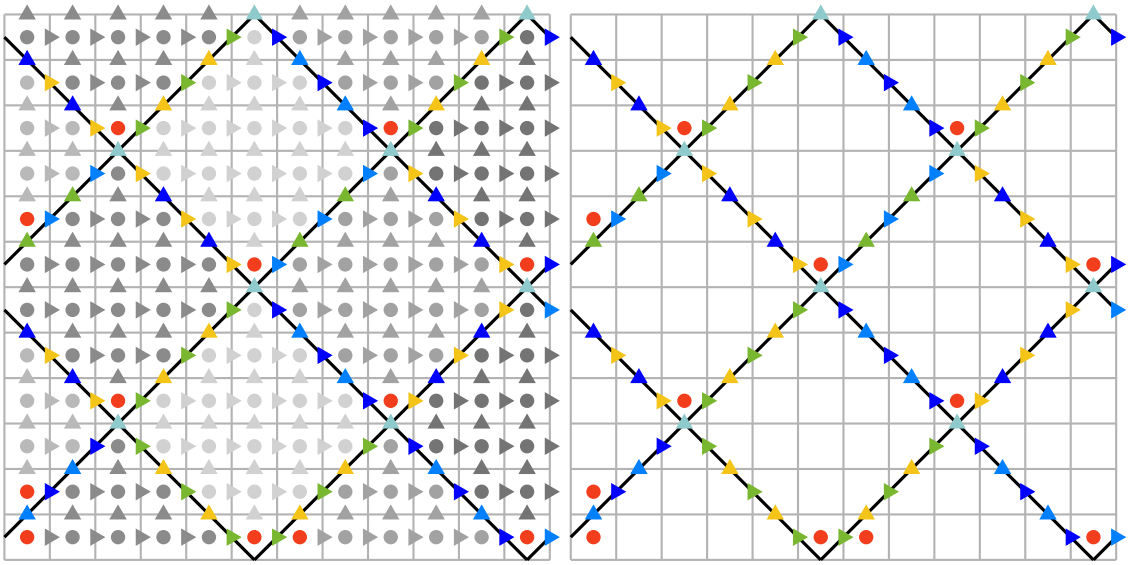
To construct a non-overlapping decomposition of the physical domain, we partition the grid geometrically by subdividing it into equally sized subdomains. The straightforward way to partition the grid is to use *standard* Cartesian partitioning. However, the intersection of horizontal and vertical separators leads to isolated pressure nodes, as shown in Fig. 2, where the isolated pressure nodes are in the top right of the subdomains. All of the surrounding velocities of these nodes are separator velocities. The cells in which these pressure nodes are located are called a *full conservation cells*. To avoid a singular matrix for the interior of the corresponding subdomain, this pressure node cannot be eliminated but instead must be retained in the Schur-complement [33]. Furthermore, the four surrounding velocity nodes in the full conservation cell should also be placed in separate groups as well, which implies that the preconditioner for the Schur complement will retain more V_Σ nodes and therefore be denser. In three dimensions, separators become planes and entire "tubes" of isolated pressure nodes occur at positions where four subdomains meet. It is even more complicated in the corners where eight subdomains meet.

The implementation of the 'retaining of full conservation cells' is intricate and prone to bugs in the code. It also leads to significant fill and a less obvious structure of the reduced V_Σ system. To prevent any occurrences of isolated pressure nodes, *skew* Cartesian separators are the better choice, which are obtained by rotating the separators in standard Cartesian partitioning by 45 degrees. For the 2D case, this is shown in Fig. 3(a). In 3D the equivalent subdomain shape turns out to be a parallelepiped, which again leads to no full conservation cells at all [30].

2.4. Multi-level method

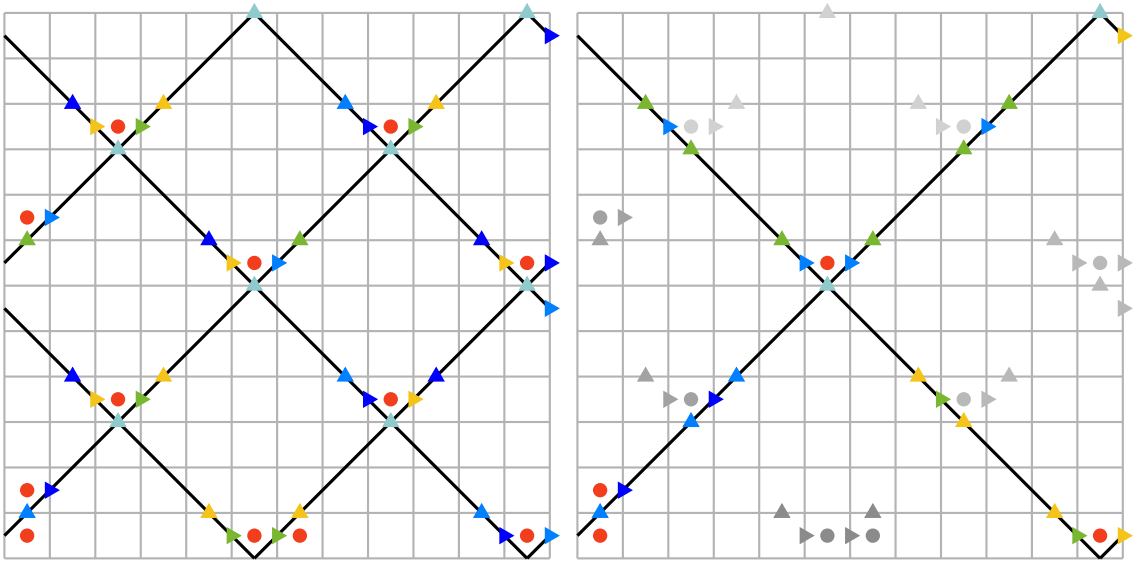
Since the V_Σ unknowns have been completely decoupled from the others by dropping, we can consider the associated matrix separately now. As mentioned, it has the same structure as the original matrix and represents the flux through the faces of our original subdomains. It is therefore natural to apply our method recursively. We introduce the *coarsening factor* (c_c) and combine in each spatial direction the V_Σ unknowns for c_c subdomains to form a subdomain of the next level. Thus, if the original subdomains had separator length $s_x = 4$ (four grid cells forming the separator), the next level will have $s_x = 8$ for $c_c = 2$, and the physical size of the new subdomains is roughly $8x$ larger in 3D. We then proceed as before: eliminate subdomain interior, form Schur-complement, transform and drop, and take the V_Σ matrix as the next level (or factor it using a sparse direct solver if it is 'small enough').

Fig. 3 depicts the process from the grid point-of-view, which shows the principle for coarsening factor $c_c = 2$. The choice of the separator length, the coarsening factor, and the number of levels determine the size of the linear systems on each level.



(a) Initial groups

(b) After elimination of interiors only the separators remain



(c) The Householder transformations leaves one V_Σ-node per separator group

(d) Domains on next level

Figure 3: Illustration of the multi-level incomplete factorization process. This example uses a coarsening factor of 2, i.e. the separators on the next level have twice the length of those on the previous level.

		Cartesian				Skew			
n_p	n_x	its	t_c	t_s	n_S	its	t_c	t_s	n_S
1	16	22	0.07	0.03	1296	19	0.11	0.03	721
4	32	23	0.34	0.07	8379	28	0.25	0.09	10368
16	64	23	1.02	0.26	76951	31	0.94	0.43	82944
128	128	24	4.61	1.03	654255	32	4.30	2.41	663552
1024	256	24	50.89	21.66	5386591	33	181.24	55.50	5308416

Table 1: 3D Poisson equation – HYMLS results for constant number of levels $L = 2$, coarsening factor $c_c = 2$ and separator length $s_x = 8$.

2.5. Convergence and complexity

In [33], it was shown that for two levels the number of iterations is independent of the mesh size. Since we are just repeating the process on the V_Σ system, it is straightforward to show that also for a fixed number of levels the number of iterations is independent of the mesh size. With an increasing number of levels the number of iterations increases, but it does so only very mildly and in a monotonous way, as our numerical experiments will show. The computational complexity depends on the number of iterations and the size of the last Schur-complement. This size increases with the problem size if the number of levels is fixed. By increasing the number of levels it drops significantly.

3. Numerical results

In this section, we will study the convergence and performance of the HYMLS preconditioner in some numerical tests. To establish a baseline, we start with the Poisson equation. After that we will introduce the lid-driven cavity problem to demonstrate the advantage of HYMLS over other approaches. The experiments were performed on the Dutch national computing facility Cartesius, of which we used the so-called thin nodes. Each of them has two 12-core sockets of Intel Xeon E5-2690 v3 (Haswell), running at 2.6GHz, and 64 GB DDR4 RAM per node.

3.1. HYMLS on the Poisson equation

For the Poisson equation $\nabla^2 u = f$, there exists an optimal algorithm, namely the multigrid method, which solves the problem in a fixed number of operations per unknown. We do not intend to compete with multigrid methods here but want to use the problem as a well understood benchmark for our own solver. The equation is solved on a cube-shaped domain with Dirichlet boundary conditions, using the second-order central finite difference discretization.

In Tables 1 and 2, we show results for a constant number of levels and constant last Schur-complement size, respectively. The columns in the table show:

- n_p: the number of cores used;
- n_x: the number of grid points per spatial dimension;
- the number of levels (L), the number of iterations (its);
- t_c: the time needed to compute the factorization;
- t_s: the time needed to solve the system to 8 digits accuracy; and
- n_S: the size of the last Schur complement.

For the sake of comparison, Table 1 shows results for both Cartesian and Skew partition partitioning. The latter is not necessary here because there is no staggered grid. We see that the number of iterations (its) is getting constant after a few refinements. This comes at the expense of a growing last Schur-complement (n_S). Since for the last Schur-complement we use a sparse direct solver, the factorization of that part dominates the factorization time (t_c) with increasing grid size, and it does also in the solution phase (t_s).

n_p	n_x	L	its	t_c	t_s	n_S	its	t_c	t_s	n_S
			Cartesian					Skew		
4	32	3	26	0.36	0.08	127	32	0.25	0.27	624
16	64	4	34	0.96	0.34	127	41	1.09	0.37	624
128	128	5	42	4.34	0.62	127	53	4.23	0.84	624
1024	256	6	49	23.40	1.26	127	65	23.25	3.09	624
16	64	3	30	0.99	0.29	1647	38	0.81	0.36	4864
128	128	4	38	3.00	0.60	1647	50	2.61	0.82	4864
1024	256	5	45	9.85	1.78	1647	61	15.08	2.58	4864

Table 2: 3D Poisson equation – HYMLS results when keeping the last Schur-complement size (n_S) constant by increasing the number of levels L. Coarsening factor is $c_c = 2$ and separator length is $s_x = 8$.

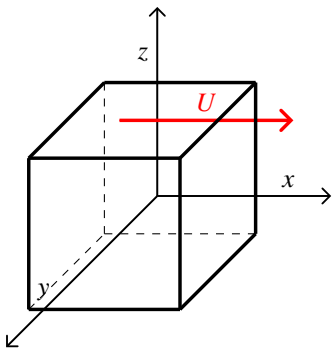


Figure 4: 3D lid-driven cavity – geometry of cavity with moving lid (speed U in x-direction) at the top.

Next, we also show some typical results of experiments where we keep the size of the last Schur-complement (n_S) fixed. Since our coarsening factor (c_c) is 2, we have to increase L by 1 on each refinement to achieve the same last Schur-complement size. We see that the number of iterations increases in the Cartesian case by 7 or 8 on refinement, and by 8 to 12 in the Skew case. So these experiments show that the number of iterations increases with the logarithm of n_x. This is an acceptable trade-off for keeping the sequential bottleneck of the last Schur complement factorization constant as the grid is refined.

Table 2 also shows that the weak scalability is not as it should be. For perfect scaling the run time should be constant for problems with n_x greater than or equal to 64. We found this to be caused by the assembly of the Schur complement matrix, a step that will be implemented more efficiently in a future version.

3.2. The 3D lid-driven cavity problem

Due to its simple geometry, the incompressible flow in lid-driven cavities plays an important role in fundamental fluid mechanics and as a numerical benchmark. We consider the flow of an incompressible Newtonian fluid in a 3D cavity with edges of length L (see Fig. 4). At the top, the fluid is driven by a lid moving with constant speed U. At all walls no-slip boundary conditions are applied, hence all boundary conditions are of Dirichlet type. The flow region is defined in the dimensionless Cartesian coordinates x, y and z, each of which varies from -0.5 to 0.5.

An important step has been made by Feldman and Gelfgat [8] who computed the critical Reynolds Number of $\mathrm{Re}_c = 1914$ for the onset of an oscillating perturbation on top of the steady flow. They found that the associated Hopf bifurcation is slightly sub-critical, i.e., for values lower than the critical value there exist two (stable) solutions.

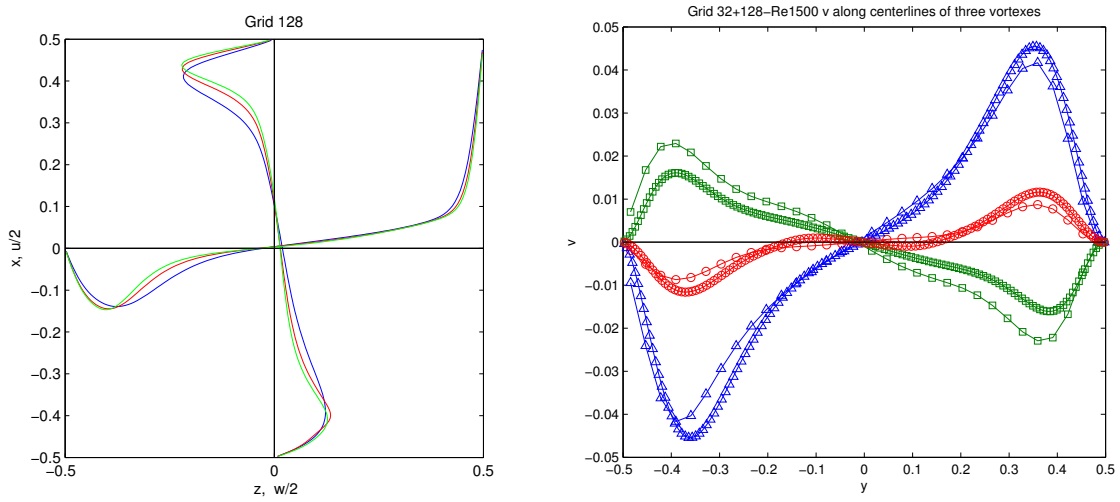


Figure 5: 3D lid-driven cavity – Steady state flow profiles obtained for grid size 128^3 and Re = 1000 (blue), Re = 1500 (red) and Re = 1900 (green). $u/2$ and $w/2$ velocity components along center lines (0,0,z) and (x,0,0), respectively (left); v component profiles for Re = 1500 along three lines going through the center of downstream secondary vortex (blue), primary vortex (green) and upstream secondary vortex (red), for grid size 32^3 and 128^3 (right).

Here, a steady one and a transient one. When Re = 1970 > Re_c, it is observed that the oscillatory flow breaks the mirror symmetry with respect to the cavity mid-plane $y = 0$. The numerical prediction of Feldman and Gelfgat is consistent with experimental investigations of Liberzon et al. [17] in 2011, who found a critical onset in the range between Re = 1700 and 1970. And it was shown that at Re = 1970, the flow exhibits oscillations characterized by a dimensionless angular frequency $\omega = 0.575$. Kuhlmann and Albensoeder [16] did very accurate computations using a spectral method and found $\mathrm{Re}_c = 1919.5$. They computed this simply by time integration of the equations and varying the Reynolds Numbers. They also studied the sub-critical behavior of this bifurcation in more detail and found that already at Re = 1921 complicated dynamics are introduced by interfering non-symmetric modes, i.e. modes that do not adopt the mirror symmetry around the plane $y = 0$. After transition to unsteadiness, with further increase of the Reynolds Number, the flow becomes turbulent.

3.3. Typical solutions

For Re ≤ 1900, the computations showed that the steady flow is the only solution. These solutions have a reflection symmetry with respect to the cavity mid-plane $y = 0$, which is shown for v in Fig. 5 (left). Note that the y-direction is orthogonal to the main circulation plane xz, see Fig. 4. It can be seen from Fig. 5 that the velocities do not change so much for Re ranging from 1000 to 1900, but their gradients become steeper and steeper close to the boundary with the increase of the Reynolds Number. Hence, a fine grid is needed in the vicinity of the walls.

Isosurfaces of oscillation amplitudes of the three velocity components are presented in Fig. 6 for developed unstable steady flow at Re = 2000. As expected, the spatial pattern of the amplitude values has a reflection symmetry with respect to the cavity mid-plane $y = 0$, where amplitudes have the largest value. The maximum amplitude is defined as $\max(\sqrt{u^2 + v^2 + w^2})$.

In our continuation program, the Reynolds Number is the continuation parameter. Starting from a small number, e.g., Re = 1, the Reynolds Number is increased with step size 500 up to Re = 1900. Meanwhile, the eigenvalues are computed in order to check the stability of the solution. The results are validated by comparing them with those of Feldman and Gelfgat [8], in which a time-dependent method is used for both 2D and 3D computations. Good agreement of the center line velocity and pressure values is observed for the relevant range of Re.

Velocities and pressures on different grids are given in Fig. 7. The difference between the results on subsequent grids is decreasing by a factor 4 as the grid is refined by a factor 2, which shows the second-order accuracy of the discretization.

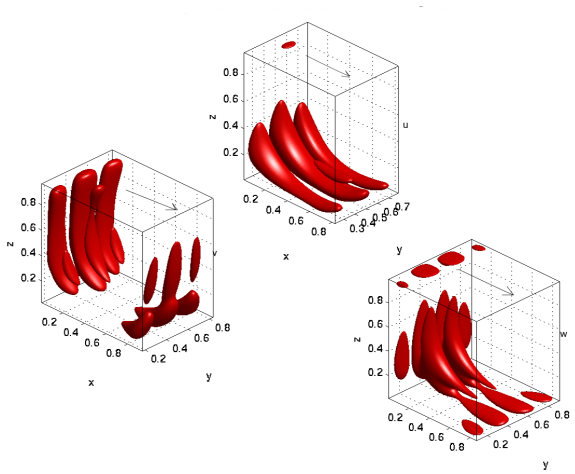


Figure 6: 3D lid-driven cavity – Isosurfaces, at 20% of the maximum velocity amplitude level, of the amplitude of the three velocity components (top u, left v, and right w) of the most unstable eigenmode at Re = 2000, grid size 128^3.

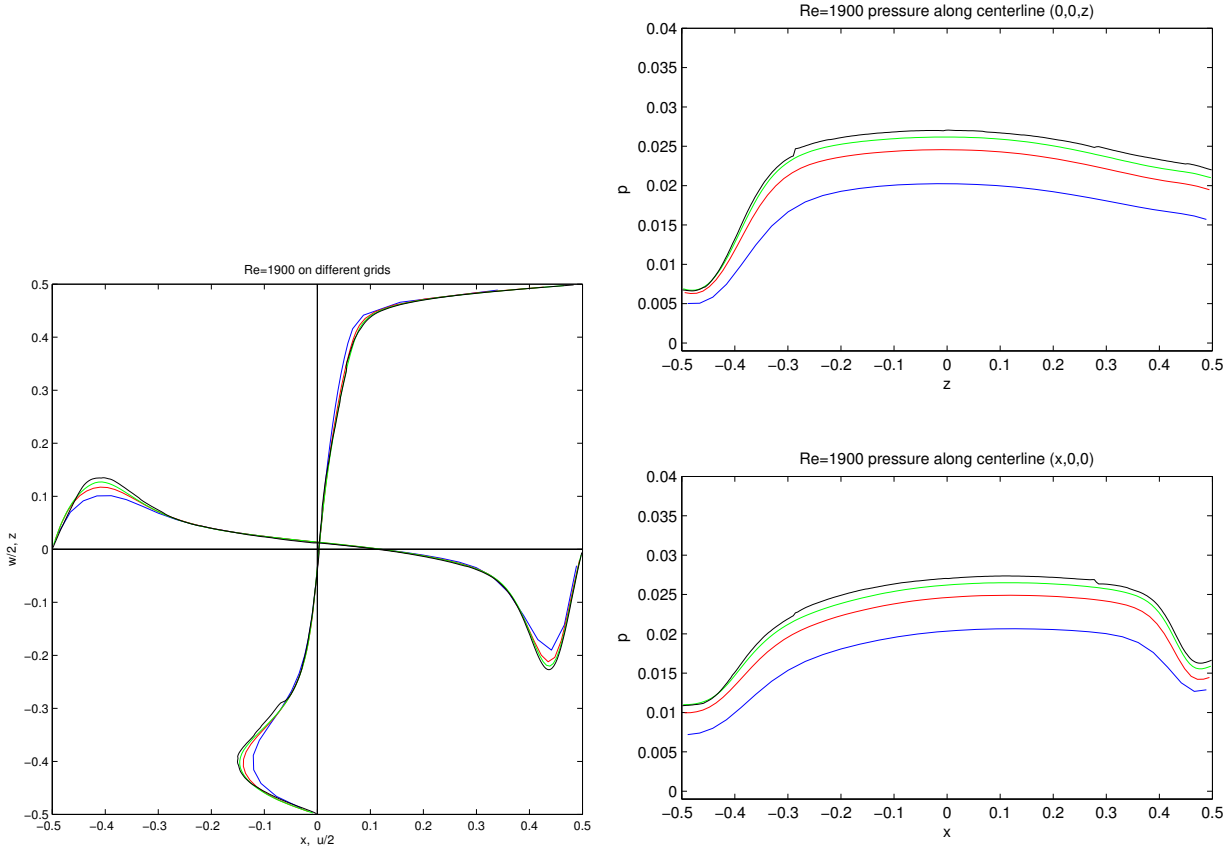


Figure 7: 3D lid-driven cavity: Velocity $u/2$ and $w/2$ components (left) and pressure p (right) along center lines (0,0,z) and (x,0,0) at Re = 1900 for grid size 32^3 (blue), 64^3 (red), 128^3 (green) and 256^3 (black).

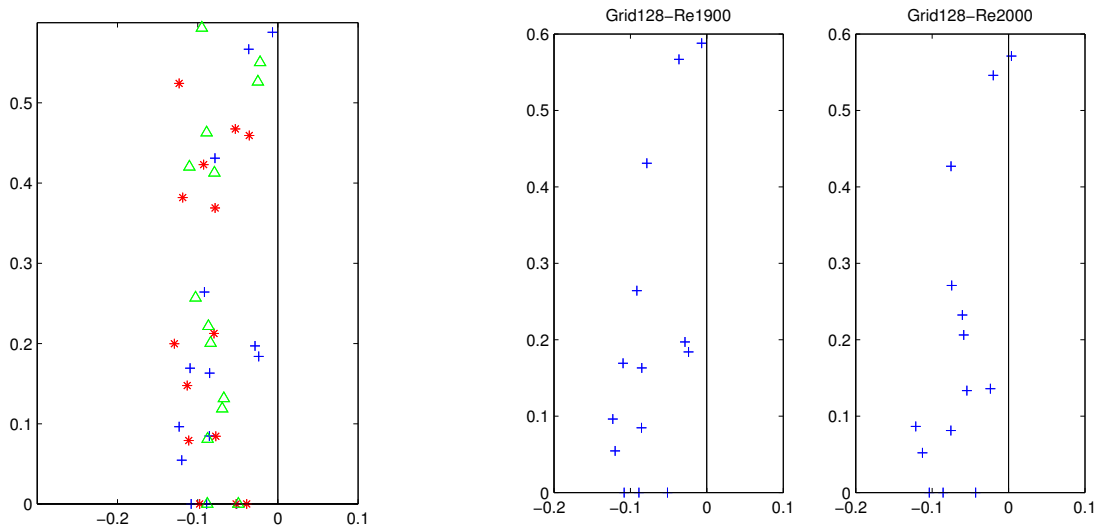


Figure 8: 3D lid-driven cavity – eigenvalues at Re=1900 for grid sizes 32^3 (red), 64^3 (green) and 128^3 (blue) (left); eigenvalues closest to zero for grid size 128^3 at Re = 1900 and Re = 2000 (right).

Re	λ_{32}	λ_{64}	λ_{128}
1900	$-53.05 + 0.468i$	$-22.28 + 0.550i$	$-6.762 + 0.588i$
2000	$-42.35 + 0.463i$	$-10.73 + 0.545i$	$3.901 + 0.571i$
Re_c	2395	2093	1963 extrapolated value 1917

Table 3: (3D lid-driven cavity) Eigenvalues obtained on different grids at Re = 1900 and Re = 2000.

There is one primary clockwise eddy in the middle and two secondary anti-clockwise eddies at the bottom of the cavity, left and right, respectively. These are the same as found in 2D [15], except that there is an additional secondary eddy at the top left of the cavity in 2D [23].

In our study, we want to use continuation of equilibria to find the critical point where the solution becomes unstable. To find the critical point, one has to compute the critical modes by solving an algebraic eigenvalue problem. Doing so, one observes that a conjugate pair of eigenvalues is crossing the imaginary axis near Re = 2000, see Fig. 8. Hence, the flow loses its stability there. Table 3 shows the eigenvalues with smallest real part on a sequence of grids. By extrapolation, based on the second-order behavior of the error, the critical Reynolds Number is expected around 1917. At that point, the eigenvalue is between $0.571i$ and $0.588i$, which is in excellent agreement with the angular frequency obtained by Liberzon et al.

Eigenvalues have been computed by the block Krylov-Schur method implemented in the Trilinos package Anasazi. To target eigenvalues near 0 we use either shift-invert or the Cayley transformation, the linear systems that result are solved using GMRES and HYMLS as a preconditioner. One eigenvalue's real part becomes positive between Re = 1900 and Re = 2000, which means there is a cyclic Hopf bifurcation occurring in this interval [10, 11].

3.4. Newton versus Picard iteration

In this section, we show in Table 4 a comparison of performance of Newton versus Picard iteration. The latter uses only the convection part, i.e. $N_1(\mathbf{u})$, in (6) instead of the full Jacobian, which makes the linear systems easier to solve due to definiteness of the operator.

Re	Newton				Picard			
	n_x=8		n_x=16		n_x=8		n_x=16	
	Nwt its.	HGMR its.	Nwt its.	HGMR its.	Picrd its.	HGMR its.	Picrd its.	HGMR its.
0	1	72	1	112	1	72	1	112
200	4	92	4	160	58	86	74	147
400	4	104	3	202	(1e-3)	97	(1e-3)	172
600	4	122	3	226				
800	3	135	3	252				
1000	3	148	3	277				

Table 4: 3D lid-driven cavity – Comparison of Newton (Nwt) and Picard (Picrd) iteration; stopping criterion for nonlinear iteration and HYMLS-preconditioned GMRES (HGMR) is 1.0e-8. In case of stagnation, the achieved residual norm is printed in parentheses.

We increased the Reynolds Number in steps of 200. One observes that the number of Newton iterations is low in this range, indicating that the behavior is not far from linear. The number of GMRES iterations gradually increases with the Reynolds Number. In contrast, the number of Picard iterations is already high for Re=200. At Re=400 the convergence stagnates around 1e-3. This occurs also at Re=300. We know that the matrix in the Picard iteration has negative eigenvalues for all Reynolds Numbers [31]; the real part of the eigenvalues is determined by the eigenvalues of the diffusion part. This makes the linear solver converge somewhat faster. It is disappointing that Picard iteration already fails at these low Reynolds Numbers, since we know that the first bifurcation is occurring much later at Re $\sim$ 1900. Around that value the complete Jacobian gets its first positive eigenvalue. In time stepping codes Picard iteration may still be useful because decreasing the step size can usually resolve the convergence problems.

3.5. Solver performance

To assess the performance of the HYMLS preconditioned solver, we run the solver for the 3D Stokes equations (Re = 0) and for Re = 500 on a series of uniform grids. Skew partitioning is used to avoid full conservation cells (Section 2.3). The stopping criterion is a reduction of the residual norm below 1e-8 compared to the norm of the right-hand side for the first Newton step at the particular Reynolds number. Tables 5 and 6 show results of Re = 0 and Re = 500, respectively, with the same quantities shown in the tables in Section 3.1; *its* is the number of GMRES iterations required in the first iteration of the Newton process. Either the number of levels (L) or the size of the last Schur complement (n_S) are kept constant.

n_p	n_x	L	its	t_c	t_s	n_S
1	16	3	107	0.55	0.52	283
4	32	4	180	2.03	3.47	283
4	32	3	176	1.98	3.37	2184
16	64	5	264	5.64	19.03	283
16	64	3	237	5.00	16.32	17176
128	128	6	381	10.54	37.37	283
128	128	3	264	6.27	34.19	136272
1024	256	7	510	58.46	55.65	283
1024	256	3	278	20.56	168.80	1085728

Table 5: 3D lid-driven cavity at Re = 0 – HYMLS results when keeping the number of levels constant or when keeping the last Schur-complement size (n_S) constant by increasing the number of levels L. Coarsening factor is $c_c = 2$ and separator length is $s_x = 8$.

The Stokes problem (Re = 0) is merely a Poisson problem restricted to the divergence-free space. For a Poisson problem, we know that without scaling or with diagonal scaling, or with ILU(0) preconditioning, we just get the n_x behavior. A symmetric Gauss-Seidel or MILU(0) preconditioning should achieve $\sqrt{n_x}$. Noting that the number of iterations doubles if we quadruple n_x we observe similar behavior here, if we keep the last Schur-complement of equal size when refining. On the other hand, if we fix the number of levels to 3, we see that the difference in number of iterations halves when we double n_x, so we see at $n_x = 256$ the 3-level method needs 278 iterations, while the difference with the previous is 14 iterations. Then the theory of geometric series says that an upper bound is reached at about 278+14=292, say 300 iterations. This is according to the theory in [28]. The convergence is independent of the number of processors for this algorithm, however, there is an upper bound of the number of processes we can use given by the number of subdomains on the finest level (which is a fairly large number in 3D, e.g. 2^{15} for $n_x = 128$ and separator length $s_x = 4$). On coarser levels, the number of active processes is reduced dynamically if needed.

Starting with the solution obtained at Re = 0 until a steady solution is reached at Re = 500, it takes 5 or 6 Newton steps to converge, while for Re = 0 one only needs 2 steps due to linearity. We consider again the 3-level method and the case where we keep the last Schur-complement of equal size when refining. In the first case, we see the number of iterations increasing to 512 at $n_x = 128$ and next it decreases slightly to 491 at $n_x = 256$. This is not accidental but due to the fact that on grid refinement the mesh Peclet-number decreases, making the viscous terms more dominant. The approximations made on the first three levels will mostly influence the diffusion terms now and hence one expects the method starting to behave more like for the Stokes problem. So on further refinement we expect to see the number of iterations decrease to approximately 300.

For the case where we keep the last Schur-complement of equal size when refining, the number of iterations goes up by a factor 3 when we quadruple n_x. If we bound that factor by 4, we get a complexity of order n_x^4 or, expressed in $N = n_x^3$, $N^{4/3}$.

n_p	n_x	L	its	t_c	t_s	n_S
1	16	3	161	0.56	0.89	283
4	32	4	342	2.05	7.90	283
4	32	3	313	2.03	7.54	2184
16	64	5	598	5.62	45.83	283
16	64	3	460	5.01	32.45	17176
128	128	6	972	10.58	99.03	283
128	128	3	512	6.34	64.70	136272
1024	256	7	1532	58.52	178.45	283
1024	256	3	491	21.09	305.50	1085728

Table 6: 3D lid-driven cavity at Re = 500 – HYMLS results when keeping the number of levels constant or when keeping the last Schur-complement size (n_S) constant by increasing the number of levels L. Coarsening factor is $c_c = 2$ and separator length is $s_x = 8$.

n_p	n_x	L	its	t_c	t_s	n_S
1	16	2	188	0.13	1.12	24
4	32	2	874	8.54	56.75	192
16	64	3	530	1307.95	634.16	48

Table 7: 3D lid-driven cavity at Re = 500 – Performance of Teko with LSC preconditioner.

In Table 7 we show results for the same Reynolds Numbers using the block preconditioner LSC (Least-Squares Commutator) implemented in Teko. We chose Teko for comparison because it is available in Trilinos and can therefore easily be coupled to our code. The stopping criterion of the linear solver is 1e-8, as before. Compared to HYMLS,

n_x	Δt (sec)	Poisson iters./time step	CPU time (sec)
64	0.05	29	660
128	0.025	35	9420
256	0.01	43	167340

Table 8: (3D lid-driven cavity) Performance of time integration at Re = 500 on different grid size.

we see that Teko has much more difficulty with the grid refinement, leading to a huge computational cost already at a grid size of 64^3. A crude computation shows that per grid point the method becomes about 30 times more expensive per iteration. This must be attributed to slow convergence of ML on the subblocks. One of these blocks is the velocity part which now contains both $N_1(\mathbf{u})$ and $N_2(\mathbf{u})$ and will be difficult to deal with by a standard AMG method. We chose the number of levels to be 2 for grid sizes 16^3 and 32^3, and 3 levels for 64^3 since these seemed to give the optimal results. Using 3 levels for 32^3 for instance caused GMRES to use more than 1500 iterations.

3.6. Comparison with a time integration method

A different approach towards computing steady solutions is by (implicit) time integration. To compare the approaches, we run the backward Euler method for the lid-driven cavity problem and the same spatial discretization and a pressure-correction scheme. The pressure Poisson equation and the momentum (transport) equations are solved separately, using MICCG and BiCGStab, respectively. That is, a modified incomplete Cholesky(0) factorization is used to precondition the Poisson equation, and no preconditioner is needed for the momentum part because the time step is relatively small (CFL number between 4 and 5). The code we used is sequential but we note that such an approach could be parallelized rather easily.

Table 8 shows the performance of this method on a 3D lid-driven cavity problem for different grid sizes, starting with the zero-solution until a steady solution is reached. For Re = 500, 50 seconds of simulated time are required to obtain the steady solution in about 8 digits. The grid in this code is exponentially stretched, with cells near the walls 4 times smaller than at the center-lines, which may have a slightly deteriorating effect on the linear solvers compared to the uniform grid used before.

Table 8 shows the performance of this method on a 3D lid-driven cavity problem on one node of Peregrine for different grid sizes, starting with the zero-solution until a steady solution is reached. Peregrine is the cluster of University of Groningen, which has 162 nodes and each node has 24 Intel Xeon E5 2680v3 2.5 GHz cores and 128 GB internal memory. The single node performance of this machine is comparable to the single node performance of Cartesius.

The convergence of MICCG behaves as expected. We see that the number of iterations is proportional to $\sqrt{n_x}$ – even lower than that, probably due to averaging over a large number of time steps in which the solution approaches a steady limit, hence the initial guesses are getting better and better. The solution time of the momentum systems is negligible because on average between 1 and 2 iterations per time step suffice. The memory consumption is increasing almost linear with the number of unknowns as expected. The CPU time increases by roughly $n_x^{9/2}$ or $N^{3/2}$. The extra $3/2$ on top of N is due to the square root increase of the number of iterations of the Poisson solver and the linear increase in the number of time steps to reach the end time (the time step Δt is roughly halved when n_x is doubled). The decrease of the time step helps to keep the number of iterations for the momentum equation low.

Comparing the time integration to the continuation, we observe that the continuation complexity is about $N^{4/3}$ at Re = 500, while that of the time stepper is about $N^{3/2}$. So the continuation has quite some overhead. Let us assume for a moment that the time integration method is perfectly parallel scalable. Then for $n_x = 256$ we need about 160 seconds with 1024 processors to get the result, where HYMLS needs about 1185 seconds in 5 Newton steps. This shows that solving with HYMLS is slower by a factor 7. However, in the time integration approach the preconditioner is an ILU(0) factorization and the solution of the linear equations dominates the turnaround time. Since ILU(0) contains data dependencies, it is not straightforward to parallelize. To overcome this, coloring has to be used which may easily give rise to more iterations and reduced performance. Both programs allow for efficiency improvements, but probably the difference would be smaller in an actual comparison with a parallel implementation. So while the

direct comparison is difficult, we believe to have shown that the approaches are competitive for the 3D lid-driven cavity problem.

4. Discussion and conclusions

We presented a holistic approach for solving the steady, incompressible Navier-Stokes equations. This includes continuation of steady states (as a globalization strategy and for bifurcation analysis), Newton linearization and the solution of the resulting sparse linear systems using our HYMLS preconditioner. The key to success is the solution of challenging linear systems using a robust multi-level approach for the PDE system discretized by a structured finite volume method. Our numerical experiments reveal a number of aspects rarely mentioned in the literature.

First we have seen that a Picard approach using the Oseen equations as an approximation to the Jacobian will only converge up to a moderate value of the Reynolds Number. In this case, it was only up to a quarter of the Reynolds for which the steady state becomes unstable. This shows that an approximation of the Jacobian requires appropriate spectral properties for the nonlinear solver to converge. In the Oseen equations the spectrum stays negative and is determined by the most critical eigenvalue of the diffusion term, as argued in the introduction, whereas a pair of eigenvalues of the Jacobian approaches the imaginary axis, giving eventually positive real parts.

The definite operator obtained in Picard iteration aids segregated preconditioners such as LSC, which use multigrid for the subsystems, however, for the full Jacobian these methods typically fail. Using HYMLS we can deal with indefiniteness since the indefiniteness will be propagated to the last Schur-complement which is tackled by a direct solver including pivoting. Here, this is shown by solving the eigenvalue problem that has been solved at Re=2000. In [33] there is also the 2D analogue discussed for the two level approach, showing its robustness far beyond the first bifurcation.

Another advantage of our fully coupled preconditioning approach is that we do not need nested iterations for the different variables (i.e. velocity and pressure). The conservation of mass is handled in a natural way because the preconditioner is tailored to the equations and discretization at hand. We believe that generalizations of the approach to mass-conservative preconditioning for other discretizations are possible.

In Section 3.6 we also compared our overall methodology to a time integration approach. A priori it is not clear whether using continuation with HYMLS will be faster than a time integration approach. A simple time integrator allows usually for a better parallelization and higher flop rate than a linear system solver. We have shown that our approach at least gets within the same range in terms of computing time, although we compared only with a sequential code. If solutions for a series of parameters are sought, the continuation approach is expected to be clearly superior, especially on fine grids.

The convergence of HYMLS for a fixed number of levels agrees with the two level approach in [33]. The number of iterations becomes constant if the size of the problem increases. This means that the preconditioned matrix has a condition number which is independent of the grid size. As the size of the last Schur-complement increases with the grid size, this step eventually becomes the bottleneck for fine grids in 3D.

The convergence of the full multilevel HYMLS, keeping the size of the last Schur complement approximately constant, on the Stokes problem did not show the mild increase of the number of iterations with n_x as is observed for the Poisson problem. This is caused by the addition of negative definite terms from the pressure to the velocity part, as is shown in [21]. However, this difficulty can be overcome. We performed first tests with a method in which the separators are approximated by more than one unknown. This is a compromise between a fixed number of levels and a fixed Schur complement size (but increasing number of levels). The hope is that the associated increase in fill is outweighed by the reduction of the number of iterations and makes the overall approach converge near grid independent while keeping the complexity proportional to the number of unknowns. Mathematical and implementation details are left to future work. Access to the HYMLS source code used in this study is provided by the authors upon request. We plan to make the source code publicly available in the future.

Acknowledgements. S.B. works within the research programme Mathematics of Planet Earth with project number 657.014.007, which is financed by the Netherlands Organisation for Scientific Research (NWO). Also the computer time on Cartesius is financed by NWO, dossier number SH-316-15.

References

[1] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numer.*, 14:1–137, 2005.

[2] M. Benzi and M. A. Olshanskii. An augmented Lagrangian-based approach to the Oseen problem. *SIAM J. Scientific Computing*, 28:2095–2113, 2006.

[3] M. A. Crisfield. *Nonlinear Finite Element Analysis of Solids and Structures, Vol 1: Basic Concepts*, chapter 1. Wiley, 1991.

[4] E. C. Cyr, J. N. Shadid, and R. S. Tuminaro. Stabilization and scalable block preconditioning for the Navier-Stokes equations. *Journal of Computational Physics*, 231(2):345 – 363, 2012.

[5] H.A. Dijkstra, F.W. Wubs, A.K. Cliffe, E. Doedel, I.F. Dragomirescu, B. Eckhardt, A.Y. Gelfgat, A. Hazel, V. Lucarini, A.G. Salinger, E.T. Phipps, J. Sanchez-Umbria, H. Schuttelaars, L. Tuckerman, and U. Thiele. Numerical bifurcation methods and their application to fluid dynamics: Analysis beyond simulation. *Communications in Computational Physics*, 15(1):2–38, 2014.

[6] H. Elman, V.E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro. A taxonomy and comparison of parallel block multi-level preconditioners for the incompressible Navier–Stokes equations. *Journal of Computational Physics*, 227(3):1790 – 1808, 2008.

[7] H. C. Elman, D. Silvester, and A. J. Wathen. Performance and analysis of saddle point preconditioners for the discrete steady-state Navier-Stokes equations. *Numerische Mathematik*, 90(4):665–688, Feb 2002.

[8] Y. Feldman and A. Y. Gelfgat. Oscillatory instability of a three-dimensional lid-driven flow in a cube. *Physics of Fluids*, 22(November 2009):1–9, 2010.

[9] C. Foias, O. Manley, R. Rosa, and R. Temam. *Navier-Stokes Equations and Turbulence*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge, 2001.

[10] M. Golubitsky and W. F. Langford. Classification and unfoldings of degenerate Hopf bifurcations. *Journal of Differential Equations*, 41(3):375 – 415, 1981.

[11] M. Golubitsky, I. Stewart, and D. G. Schaeffer. *Singularities and Groups in Bifurcation Theory. Vol. II*, chapter 2. Applied Mathematical Sciences, vol. 69. Springer, New York, 1988.

[12] Nguyenho Ho, Sarah D. Olson, and Homer F. Walker. Accelerating the Uzawa algorithm. *SIAM J. Sci. Comput.*, 39(5), 2018.

[13] D. Kay, D. Loghin, and A. Wathen. A preconditioner for the steady-state Navier-Stokes equations. *SIAM Journal on Scientific Computing*, 24(1):237–256, 2002.

[14] H. B. Keller. Numerical solution of bifurcation and nonlinear eigenvalue problems. *Applications of bifurcation theory*, 1(38):359–384, 1977.

[15] H. C. Ku, R. S. Hirsh, and T. D. Taylor. A pseudospectral method for solution of the three-dimensional incompressible Navier-Stokes equations. *Journal of Computational Physics*, 70(2):439 – 462, 1987.

[16] H. C. Kuhlmann and S. Albensoeder. Stability of the steady three-dimensional lid-driven flow in a cube and the supercritical flow dynamics. *Physics of Fluids*, 26, 2014.

[17] A. Liberzon, Y. Feldman, and A. Y. Gelfgat. Experimental observation of the steady-oscillatory transition in a cubic lid-driven cavity. *Physics of Fluids*, 23:1–14, 2011.

[18] K. Lust and D. Roose. *Computation and Bifurcation Analysis of Periodic Solutions of Large-Scale Systems*, pages 265–301. Springer New York, New York, NY, 2000.

[19] K. Meerbergen. An implicitly restarted rational Krylov strategy for Lyapunov inverse iteration. *IMA Journal of Numerical Analysis*, 36(2):655–674, 2016.

[20] K. Meerbergen and A. Spence. Shift-and-invert iteration for purely imaginary eigenvalues with application to the detection of Hopf bifurcations in large scale problems. *SIAM J. Matrix Anal. Appl.*, 31:1463–1482, 01 2010.

[21] A. C. De Niet and F. W. Wubs. Two saddle point preconditioners for fluid flows. *Int. J. Numer. Methods Fluids*, 54:355–377, 2007.

[22] A. C. De Niet and F. W. Wubs. Numerically stable LDL^T-factorization of F-type saddle point matrices. *IMA Journal of Numerical Analysis*, 29:208–234, 2009.

[23] K. Poochinapan. Numerical implementations for 2D lid-driven cavity flow in stream function formulation. *ISRN Applied Mathematics*, 2:536 – 558, 2012.

[24] Y. Saad. *Numerical Methods for Large Eigenvalue Problems, Revised Edition*, volume 66, pages 128–134. Siam, 2011.

[25] A. Segal, M. Rehman, and C. Vuik. Preconditioners for incompressible Navier-Stokes solvers. *Numerical Mathematics-theory Methods and Applications*, 3, 08 2010.

[26] G. M. Shroff and H. B. Keller. Stabilization of unstable procedures: The recursive projection method. *SIAM Journal on Numerical Analysis*, 30(4):1099–1120, 1993.

[27] G. L. G. Sleijpen and H. A. Van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications*, 17(2):401–425, 1996.

[28] J. Thies and F. W. Wubs. Design of a parallel hybrid direct/iterative solver for CFD problems. In *Proceedings of the 2011 IEEE Seventh International Conference on eScience*, ESCIENCE '11, pages 387–394, Washington, DC, USA, 2011. IEEE Computer Society.

[29] G. Tiesinga, F.W. Wubs, and A.E.P. Veldman. Bifurcation analysis of incompressible flow in a driven cavity by the Newton–Picard method. *Journal of Computational and Applied Mathematics*, 140(1):751 – 772, 2002. Int. Congress on Computational and Applied Mathematics 2000.

[30] M. van der Klok. *Skew partitioning for the hybrid multilevel solver*, pages 19–22. Bachelor thesis, University of Groningen, 2017.

[31] R. W. C. P. Verstappen and A. E. P. Veldman. Symmetry-preserving discretization of turbulent flow. *J. Comput. Phys.*, 187(1):343–368, May 2003.

[32] J. B. Vos, A. Rizzi, D. Darracq, and E. H. Hirschel. Navier-Stokes solvers in European aircraft design. *Progress in Aerospace Sciences*, 38(8):601 – 697, 2002.

[33] F. W. Wubs and J. Thies. A robust two-level incomplete factorization for (Navier-) Stokes saddle point matrices. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1475–1499, 2011.