SIMPLE-type preconditioners for the Oseen problem

C. Vuik, M. ur Rehman, and A. Segal

Delft University of Technology

Delft Institute of Applied Mathematics

Delft, The Netherlands.

Outline

- Introduction
- Solution technique
- IDR(s) method
- Preconditioning
- Numerical experiments
- Conclusions

The incompressible Navier Stokes equation

$$-\nu\nabla^2\mathbf{u} + \mathbf{u}.\nabla\mathbf{u} + \nabla p = f \quad \text{in} \quad \Omega$$

$$\nabla.\mathbf{u} = 0 \quad \text{in} \quad \Omega.$$

$\mathbf{u}$ is the fluid velocity vector

p is the pressure field

$\nu > 0$ is the kinematic viscosity coefficient ($1/Re$).

$\Omega \subset \mathbf{R}^{2 \text{ or } 3}$ is a bounded domain with the boundary condition:

$$\mathbf{u} = \mathbf{w} \ \text{ on } \ \partial\Omega_D, \quad \nu\frac{\partial\mathbf{u}}{\partial\mathbf{n}} - \mathbf{n}p = 0 \ \text{ on } \ \partial\Omega_N.$$

Linear system

The finite element discretization give rise to a non-linear system.
Matrix form after linearization:

$$\begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}$$

where $F \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{m \times n}$, $f \in \mathbb{R}^n$ and $m \leq n$

- $F = A$ in Stokes problem, A is vector Laplacian matrix
- $F = \nu A + N$ in Picard linearization, N is vector-convection matrix
- $F = \nu A + N + W$ in Newton linearization, W is the Newton derivative matrix
- B is the divergence matrix

Sparse linear system, Symmetric indefinite (Stokes problem), nonsymmetric otherwise.
Saddle point problem having large number of zeros on the main diagonal

Iterative Solution Techniques

- **Classical Iterative Schemes:**
  Methods based on matrix splitting, generates sequence of iterations
  $x_{k+1} = M^{-1}(Nx_k + b) = Qx_k + s$, where $\mathcal{A} = M - N$
  Jacobi, Gauss Seidel, SOR, SSOR

- **Krylov Subspace Methods:**
  $x_{k+1} = x_k + \alpha_k p_k$
  Some well known methods are
  CGNR[1975], QMR[1991], CGS[1989], Bi-CGSTAB[1992], GMRES[1986], GMRESR[1994], GCR[1986], IDR(s)[2007]

IDR and IDR(s) (Induced Dimension Reduction)

Sonneveld developed IDR the 1970's. IDR is a finite termination Krylov method for solving nonsymmetric linear systems.

Analysis showed that IDR can be viewed as Bi-CG combined with linear minimal residual steps.

This discovery led to the development of first CGS, and later of Bi-CGSTAB (by van der Vorst).

As a result of these developments the basic IDR-idea was abandoned for the Bi-CG-approach.

Recently, Sonneveld and van Gijzen discovered that the IDR-approach was abandoned too soon and proposed a generalization of IDR: IDR(s).

The IDR approach for solving $Ax = b$

Generate residuals $\boldsymbol{r}_n = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_n$ that are in subspaces $\mathcal{G}_j$ of decreasing dimension.

These nested subspaces are related by

$$\mathcal{G}_j = (\boldsymbol{I} - \omega_j \boldsymbol{A})(\mathcal{G}_{j-1} \cap \mathcal{S})$$

where

- $\mathcal{S}$ is a fixed proper subspace of $\mathbb{C}^N$. $\mathcal{S}$ can be taken to be the orthogonal complement of s randomly chosen vectors $\boldsymbol{p}_i, i = 1 \cdots s$.
- The parameters $\omega_j \in \mathbb{C}$ are non-zero scalars.

It can be proved that ultimately $\boldsymbol{r}_n \in \{\boldsymbol{0}\}$ (IDR theorem).

IDR versus Bi-CG

The IDR(s) forces the residual to be in an increasingly small subspace, while Bi-CG constructs a residual in an increasingly large subspace. Yet, IDR(s) is closely related to:

- Bi-CGSTAB: IDR(1) and Bi-CGSTAB are mathematically equivalent.
- ML(k)BiCGSTAB (Yeung and Chan, 1999): This method generalizes Bi-CGSTAB using multiple 'shadow residuals'. Mathematically IDR(s) and ML(k)BiCGSTAB differ in the selection of the parameters ω_j.
  IDR(s) uses simpler recurrences, less vector operations and memory than ML(k)BiCGSTAB, and is more flexible (e.g. to avoid break down).

Prototype IDR(s) algorithm.

while $\|\boldsymbol{r}_n\| > TOL$ **or** $n < MAXIT$ **do**

for $k = 0$ to s **do**

Solve $\boldsymbol{c}$ from $\boldsymbol{P}^H d\boldsymbol{R}_n \boldsymbol{c} = \boldsymbol{P}^H \boldsymbol{r}_n$

$\boldsymbol{v} = \boldsymbol{r}_n - d\boldsymbol{R}_n \boldsymbol{c}$; $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{v}$;

if $k = 0$ **then**

$\omega = (\boldsymbol{t}^H \boldsymbol{v})/(\boldsymbol{t}^H \boldsymbol{t})$;

end if

$d\boldsymbol{r}_n = -d\boldsymbol{R}_n \boldsymbol{c} - \omega \boldsymbol{t}$; $d\boldsymbol{x}_n = -d\boldsymbol{X}_n \boldsymbol{c} + \omega \boldsymbol{v}$;

$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n + d\boldsymbol{r}_n$; $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + d\boldsymbol{x}_n$;

$n = n + 1$;

$d\boldsymbol{R}_n = (d\boldsymbol{r}_{n-1} \cdots d\boldsymbol{r}_{n-s})$; $d\boldsymbol{X}_n = (d\boldsymbol{x}_{n-1} \cdots d\boldsymbol{x}_{n-s})$;

end for

end while

More information

More information: http://ta.twi.tudelft.nl/nw/users/gijzen/IDR.html

- IDR(s) is described in: IDR(s): a family of simple and fast algorithms for solving large nonsymmetric linear systems. (To appear in revised version in SISC).

- The relation of IDR(s) with Bi-CGSTAB, and how to derive generalizations of Bi-CGSTAB using IDR-ideas can be found in: Bi-CGSTAB as an induced dimension reduction method (with Sleijpen).
- A high quality IDR(s) implementation is described in: An elegant IDR(s) variant that efficiently exploits bi-orthogonality properties.
- MATLAB implementation of IDR(s).

Preconditioning

A linear system $\mathcal{A}x = b$ is transformed into $P^{-1}\mathcal{A}x = P^{-1}b$ such that

- $P \approx \mathcal{A}$
- Eigenvalues of $P^{-1}\mathcal{A}$ are more clustered than $\mathcal{A}$
- $Pz = r$ cheap to compute

Several approaches, we will discuss here

- Block triangular preconditioners
  (LSC, Least Squares Commutator)
- SIMPLE-type block preconditioners
- Preconditioners comparison (with SILU[Rehman2008])
- Preconditioned IDR(s) and Bi-CGSTAB comparison

Block preconditioners

Block triangular preconditioner

$$\begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} = \begin{bmatrix} I & 0 \\ BF^{-1} & I \end{bmatrix} \underbrace{\begin{bmatrix} F & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & F^{-1}B^T \\ 0 & I \end{bmatrix}}$$

$$P_t = \begin{bmatrix} F & B^T \\ 0 & S \end{bmatrix}, \; S = -BF^{-1}B^T \text{(Schur complement matrix)}$$

Subsystems: solve z_2 from $Sz_2 = r_2$, and z_1 from $Fz_1 = r_1 - B^T z_2$

Block preconditioners

Generalized eigenvalue problem

$$\begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \lambda \begin{bmatrix} F & B^T \\ 0 & S \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix},$$

This eigenvalue problem has the eigenvalues $\lambda = 1$ of multiplicity n and the remaining eigenvalues depend on the Schur complement

$$BF^{-1}B^T p = \mu_i S p,$$

$\mu_i = 1$ if $S = BF^{-1}B^T$, however

- In practice F^{-1} and S^{-1} are expensive.
- F^{-1} is obtained by an approximate solve
- S is first approximated and then solved inexactly

Block preconditioners

<u>Least squares commutator (LSC) preconditioner</u>

[Elman, Howle, Shadid, Silvester and Tuminaro, 2002]

$$S \approx -(BQ^{-1}B^T)(BQ^{-1}FQ^{-1}B^T)^{-1}(BQ^{-1}B^T)$$

Q is the diagonal of the velocity mass matrix.

- Two Poisson solves
- One velocity solve

SIMPLE(R) preconditioner

$$\begin{pmatrix} u^* \\ p^* \end{pmatrix} = \begin{pmatrix} u^k \\ p^k \end{pmatrix} + M_L^{-1} B_L \left(\begin{pmatrix} r_u \\ r_p \end{pmatrix} - A \begin{pmatrix} u^k \\ p^k \end{pmatrix} \right),$$

$$\begin{pmatrix} u^{k+1} \\ p^{k+1} \end{pmatrix} = \begin{pmatrix} u^* \\ p^* \end{pmatrix} + B_R M_R^{-1} \left(\begin{pmatrix} r_u \\ r_p \end{pmatrix} - A \begin{pmatrix} u^* \\ p^* \end{pmatrix} \right).$$

Where

$$B_R = \begin{pmatrix} I & -D^{-1}B^T \\ 0 & I \end{pmatrix}, \; M_R = \begin{pmatrix} F & 0 \\ B & \hat{S} \end{pmatrix} \text{ and}$$

$$B_L = \begin{pmatrix} I & 0 \\ -BD^{-1} & I \end{pmatrix}, \; M_L = \begin{pmatrix} F & B^T \\ 0 & \hat{S} \end{pmatrix}.$$

SIMPLE-type preconditioner

Assuming u^* and p^* equal zero, the steps in SIMPLE reduce to:

SIMPLE preconditioner[Vuik 2000]:

1. Solve $Fu^* = r_u$.
2. Solve $\hat{S}\delta p = r_p - Bu^*$.
3. update $u = u^* - D^{-1}B^T\delta p$.
4. update $p = \delta p$.

- One Poisson solve
- One velocity solve

SIMPLE-type preconditioner

Assuming u^k and p^k equal zero, the steps in SIMPLER reduce to:

SIMPLER preconditioner:

1. Solve $\hat{S}p^* = r_p - BD^{-1}r_u$
2. Solve $Fu^* = r_u - B^T p^*$.
3. Solve $\hat{S}\delta p = r_p - Bu^*$.
4. update $u = u^* - D^{-1}B^T \delta p$.
5. update $p = p^* + \delta p$.

Lemma: In the SIMPLER preconditioner/algorithm, both variants (one or two velocity solves) are identical.

- Two Poisson solve
- One velocity solve
- Gives faster convergence than SIMPLE

Improvements in SIMPLE-type preconditioners

- Relaxation parameter
- hSIMPLER
- MSIMPLER

Improvements in SIMPLE(R) preconditioners

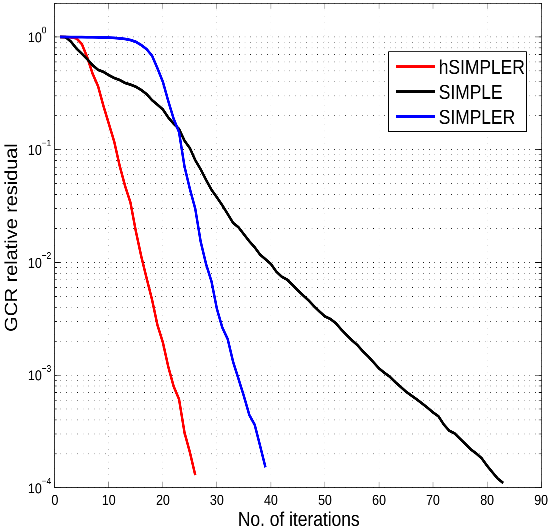
<u>Relaxation parameter:</u>

- Under-relaxation is well-known in SIMPLE-type methods.
- In SIMPLE preconditioner, velocity relaxation has no effect on the convergence, therefore only pressure is under-relaxed by a factor ω.
  $p = p^* + \omega\delta p$, where ω is chosen between 0 and 1.
- ω has no effect on convergence with SIMPLER due to extra pressure correction step.
- Faster convergence is achieved in some cases.
- Choice of ω is currently based on trial an error.

Improvements in SIMPLE(R) preconditioners

hSIMPLER preconditioner:

In hSIMPLER (hybrid SIMPLER), first iteration of Krylov method preconditioned with SIMPLER is done with SIMPLE and SIMPLER is employed afterwards.



- Faster convergence than SIMPLER
- Effective in the Stokes problem

TU Delft

Improvements in SIMPLE(R) preconditioners

MSIMPLER preconditioner:

Making the following changes in SIMPLER leads to the MSIMPLER preconditioner.

LSC: $\hat{S} \approx -(B\hat{Q_u}^{-1}B^T)(B\hat{Q_u}^{-1}\underbrace{F\hat{Q_u}^{-1}}B^T)^{-1}(B\hat{Q_u}^{-1}B^T)$

assuming $F\hat{Q_u}^{-1} \approx I$ (time dependent problems with a small time step)

$\hat{S} = -B\hat{Q_u}^{-1}B^T$

MSIMPLER uses this approximation for the Schur complement and updates scaled with $\hat{Q_u}^{-1}$.

-Convergence better than other variants of SIMPLE

-Cheaper than SIMPLER (in construction) and LSC (per iteration)

Numerical Experiments

- Driven Cavity flow (2D)
- Backward facing step flow (2D and 3D)
- Q2-Q1 finite element discretization [Taylor, Hood - 1973]
- Q2-P1 finite element discretization [Crouzeix, Raviart - 1973]
- GCR(20), Bi-CGSTAB, GMRES, IDR(s)
- The iteration is stopped if the linear systems satisfy $\frac{\|r^k\|_2}{\|b\|_2} \leq tol$,
- Experiments done with IFISS (Matlab program) and SEPRAN (industrial FEM code)

TU Delft

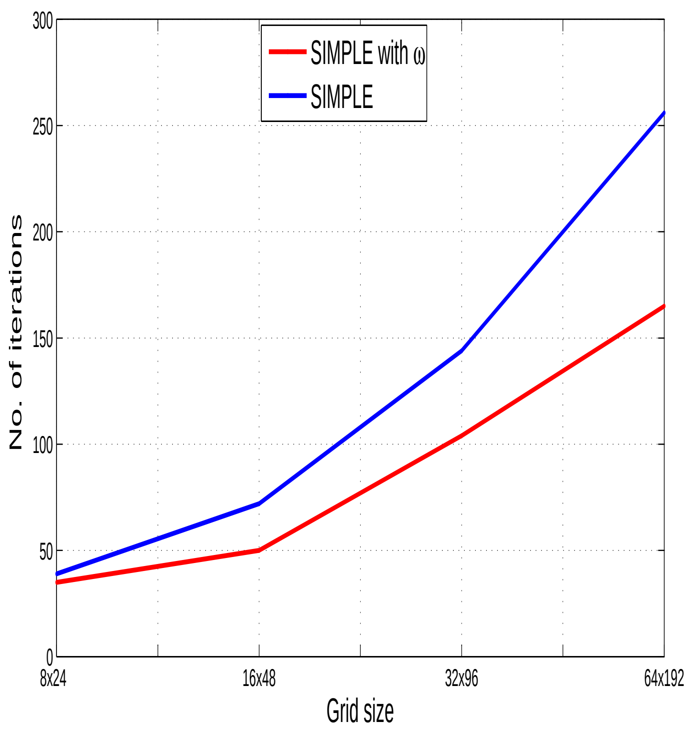
Numerical Experiments (SIMPLE type preconditioners)

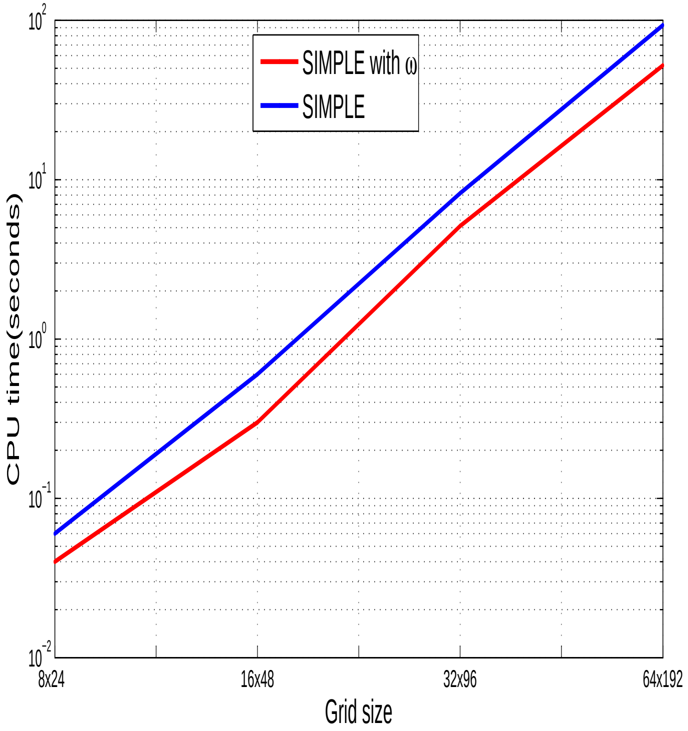
Stokes backward facing step solved with preconditioned GCR(20) with $accuracy$ of 10^{-6} , PCG used as an inner solver (SEPRAN), Green: Low inner accuracy , Yellow: High inner accuracy

Grid	SIMPLE	SIMPLER	hSIMPLER	MSIMPLER
	iter. (ts)	iter. (ts)	iter. (ts)	iter. (ts)
8×24	39(0.06)	26(0.05)	19(0.03)	11(0.02)
	37(0.14)	19(0.07)	17(0.06)	12(0.05)
16×46	72(0.6)	42(0.5)	31(0.34)	12(0.1)
	68(1.94)	30(0.86)	24(0.68)	15(0.44)
32×96	144(8.2)	NC	44(5.97)	16(0.9)
	117(34)	114(32)	37(10.6)	20(5.75)
64×192	256(93)	NC	89(141)	23(8.5)
	230(547)	NC	68(161)	25(60)

Numerical Experiments (SIMPLE type preconditioners)

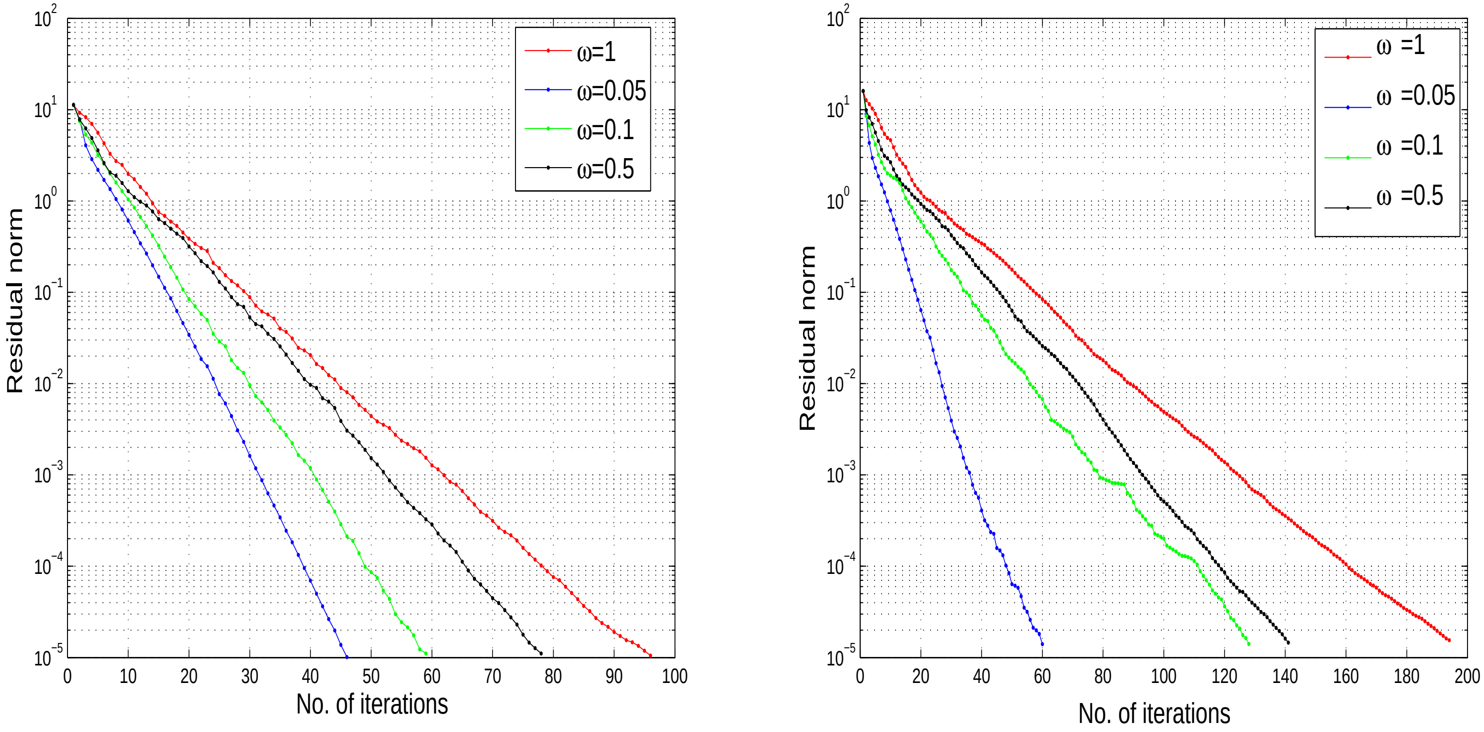
SIMPLE with relaxation parameter





Numerical Experiments (SIMPLE type preconditioners)

Effect of relaxation parameter: The Stokes problem solved in Q2-Q1 discretized driven cavity problem with varying ω: 32×32 grid (Left), 64×64 grid (Right).



TU Delft

Numerical Experiments (overall comparison)

3D Backward facing step: Preconditioners used in the Stokes problem with preconditioned GCR(20) with $accuracy$ of 10^{-6} (SEPRAN) using Q2-Q1 hexahedrons

Grid	SIMPLE	LSC	MSIMPLER
iter. $(t_s)\frac{\text{in-it-}u}{\text{in-it-}p}$			
$8 \times 8 \times 16$	44(4) $\frac{97}{342}$	16(1.9) $\frac{41}{216}$	14(1.4) $\frac{28}{168}$
$16 \times 16 \times 32$	84(107) $\frac{315}{1982}$	29(51) $\frac{161}{1263}$	17(21) $\frac{52}{766}$
$24 \times 24 \times 48$	99(447) $\frac{339}{3392}$	26(233) $\frac{193}{2297}$	17(77) $\frac{46}{1116}$
$32 \times 32 \times 40$	132(972) $\frac{574}{5559}$	37(379) $\frac{233}{2887}$	20(143) $\frac{66}{1604}$

Numerical Experiments (overall comparison)

3D Backward facing step: Preconditioners used in solving the Navier-Stokes problem with preconditioned GCR(20) with $accuracy$ of 10^{-2} (SEPRAN) using Q2-Q1 hexahedrons

Re	LSC	MSIMPLER	SILU
	GCR iter. (t_s)	GCR iter. (t_s)	Bi-CGSTAB iter. (t_s)
		$16 \times 16 \times 32$	
100	173(462)	96(162)	321(114)
200	256(565)	145(223)	461(173)
400	399(745)	235(312)	768(267)
		$32 \times 32 \times 40$	
100	240(5490)	130(1637)	1039(1516)
200	NC	193(2251)	1378(2000)
400	675(11000)	295(2800)	1680(2450)

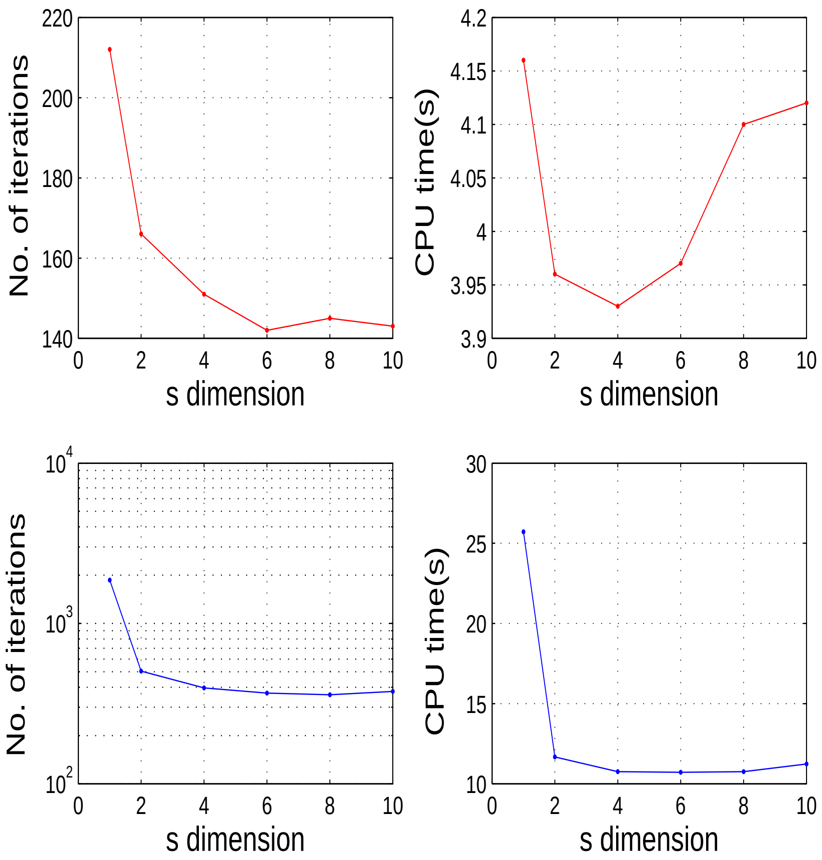
Numerical Experiments (overall comparison)

3D Lid driven cavity problem (tetrahedrons):The Navier-Stokes problem is solved with accuracy 10^{-4}, a linear system at each Picard step is solved with accuracy 10^{-2} using preconditioned Krylov subspace methods. Bi-CGSTAB is used as inner solver in block preconditioners(SEPRAN)

Re	LSC	MSIMPLER	SILU
	GCR iter. (t_s)	GCR iter. (t_s)	Bi-CGSTAB iter. (t_s)
	$16 \times 16 \times 16$		
20	30(20)	20(16)	144(22)
50	57(37)	37(24)	234(35)
100	120(81)	68(44)	427(62)
	$32 \times 32 \times 32$		
20	38(234)	29(144)	463(353)
50	87(544)	53(300)	764(585)
100	210(1440)	104(654)	1449(1116)

Numerical Experiments (IDR(s))

IDR(s): Top: 32×32, Bottom: 64×64 driven cavity Stokes flow problem



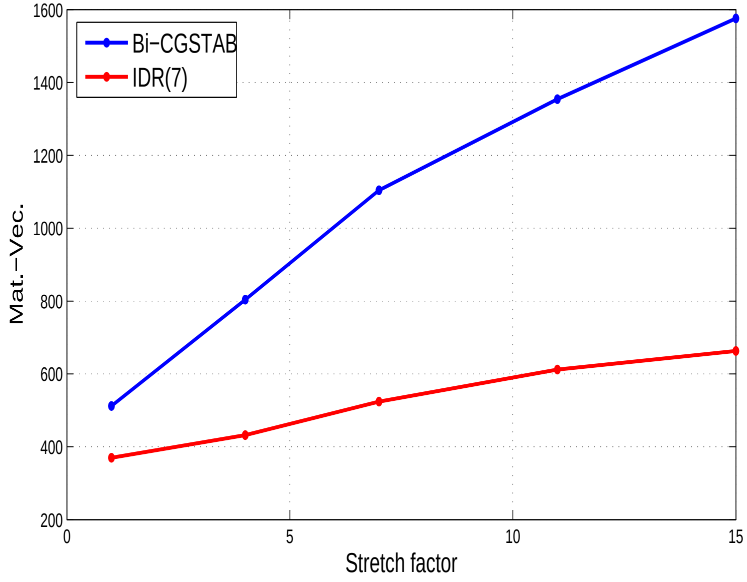
Numerical Experiments (IDR(s) vs Bi-CGSTAB)

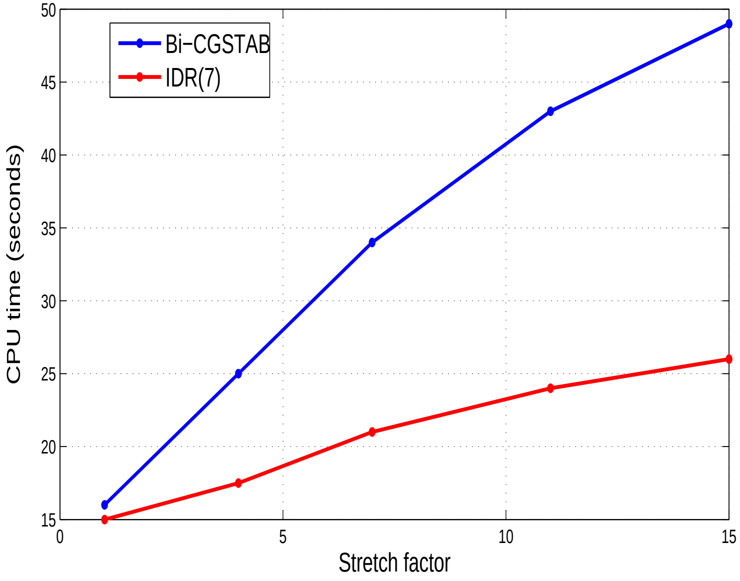
SILU preconditioner: Comparison of iterative methods for increasing grid size for the driven cavity Stokes flow problem.

Grid	Bi-CGSTAB	IDR(4)
	Mat.-Vec. (ts)	Mat.-Vec. (ts)
16×16	38(0.01)	33(0.01)
32×32	90(0.14)	75(0.14)
64×64	214(1.6)	159(1.4)
128×128	512(16)	404(15)
256×256	1386(183)	1032(156)

Numerical Experiments (IDR(s) vs Bi-CGSTAB)

SILU preconditioned: Comparison of iterative methods for increasing stretch factor for the driven cavity Stokes problem.





Numerical Experiments (IDR(s) vs Bi-CGSTAB)

SILU preconditioned: Comparison of iterative methods for the backward facing step Stokes problem.

Grid	Bi-CGSTAB	IDR(s)	
	Mat.-Vec.(ts)	Mat.-Vec.(ts)	s
32×96	214(1.3)	168(1.26)	4
64×96	NC	597(7.7)	4
96×96	NC	933(18)	4
128×96	NC	1105(31)	8

Conclusions

- Relaxation parameter improves performance of the SIMPLE preconditioner.
- hSIMPLER shows faster convergence than SIMPLER.
- MSIMPLER is at present the fastest of all SIMPLE-type preconditioners.
- In contrast with SIMPLER and hSIMPLER , SIMPLE and MSIMPLER are not sensitive to the accuracies that are used for the inner solvers.
- In all our experiments MSIMPLER proved to be cheaper than LSC. This concerns both the number of outer iterations, inner iterations and CPU time.
- In our experiments, MSIMPLER proved to be cheaper than SILU, especially when the problem is solved with high accuracy.
- IDR(s) is faster and more robust than Bi-CGSTAB.

References

- ⋆ C. Vuik and A. Saghir and G.P. Boerstoel, "The Krylov accelerated SIMPLE(R) method for flow problems in industrial furnaces," *International Journal for Numerical methods in fluids*, 33 pp. 1027-1040, 2000.
- ⋆ M. ur Rehman and C. Vuik and G. Segal, "A comparison of preconditioners for incompressible Navier-Stokes solvers," *International Journal for Numerical methods in fluids*, 57, pp. 1731-1751, 2008.
- ⋆ M. ur Rehman and C. Vuik and G. Segal, "SIMPLE-type preconditioners for the Oseen problem," *International Journal for Numerical methods in fluids*, To appear.
- ⋆ Peter Sonneveld and Martin B. van Gijzen, "IDR(s): a family of simple and fast algorithms for solving large nonsymmetric linear systems," *SIAM J. Sci. Comput.*, To appear.