# Low-Rank Update of the Restricted Additive Schwarz Preconditioner for Nonlinear Systems

Laurent Berenguer[1] and Damien Tromeur-Dervout[1]

**Key words:** Quasi-Newton Methods ; Restricted Additive Schwarz ; Preconditioner

We consider the solution of differential equations of the form Eq.(1) for a given initial condition $y(0) = y_0$ and suitable boundary conditions.

$$M\dot{y} = g(y,t) \tag{1}$$

In Equation (1), $g \in C^1(\Omega, \mathbb{R}^n)$ , for $\Omega$ an open set in $\mathbb{R}^n \times \mathbb{R}^*$ and $M \in \mathbb{R}^{n\times n}$. This equation is called a linear differential-algebraic equation (DAE) if the matrix $M$ is singular. The time discretization of Eq. (1) via backward differentiation formulas leads to solving a system of nonlinear equations $f(y) = 0$ for $f : \mathbb{R}^n \to \mathbb{R}^n$ at each time step. These equations are generally solved by Newton-like methods which require the solution of numerous linear systems of the form:

$$J_k \Delta x_k = -f(x_k) \tag{2}$$

where $J_k \in \mathbb{R}^{n\times n}$ is the Jacobian matrix of $f$ at $x_k$, or an approximation of it. In this paper we deal with the solution of these linear systems by a parallel Krylov iterative method. The condition number of the matrix $J_k$ can be very large, hence, a good preconditioner is required.

Preconditioners based on the additive Schwarz method are often used to precondition sparse linear systems. The combination of a Newton method with a Krylov method preconditioned by a Schwarz method is generally called Newton-Krylov-Schwarz [5] and has widely be applied to CFD problems (see for example [6, 14, 7]). In this paper we deal with the Restricted Additive Schwarz preconditioner [8]. Computing and solving such linear systems is generally the most time consuming part of ODE/DAE integration codes, even if there are usually only slight changes between two consecutive linear systems. When the analytic Jacobian matrix is not available, a finite difference scheme is commonly used to approximate it [12] or its matrix-vector product [15]. Another way to avoid the computation of the Jacobian matrix is to update it from one iteration to another using quasi-Newton methods [10] that converge superlinearly [4]. Since Krylov methods are used to solve Equation (2), providing a preconditioner is a critical point. A balance must be found between the ability of the preconditioner to reduce the number of Krylov iterations, and its computational cost. Then, one may want to update the preconditioner using the se-

[1] Université de Lyon, Université Lyon 1, CNRS, UMR 5208 Institut Camille Jordan, 43 boulevard du 11 novembre 1918, 69622 Villeurbanne cedex, France, e-mail: {laurent.berenguer}{damien.tromeur-dervout}@univ-lyon1.fr

cant condition in order to improve its efficiency. This idea is not new, and has been widely discussed in [3, 2]. The aim of this paper is to extend these techniques to domain decomposition based preconditioners such as the Restricted Additive Schwarz preconditioner. First, we present the Broyden update and its application to general preconditioners. Then we discuss the practical issues in applying this update to the RAS preconditioner. The third part is devoted to numerical experiments on the CFD problem of the lid-driven cavity.

## 1 The Update of the RAS Preconditioner

The preconditioned linear system of the Newton iterations can be written as:

$$G_k J(x_k)\Delta x_k = -G_k f(x_k) \tag{3}$$

or

$$J(x_k)G_k G_k^{-1}\Delta x_k = -f(x_k) \tag{4}$$

depending on which side the preconditioner is applied.

For the sake of simplicity, we use the notations $f_k = f(x_k)$ and $\Delta f_k = f_{k+1} - f_k$ in the following. The quasi-Newton update of $G_k$, that satisfies the secant condition:

$$\Delta x_k = G_{k+1}\Delta f_k \tag{5}$$

is given by:

$$G_{k+1} = G_k + (\Delta x_k - G_k \Delta f_k)\frac{v_k^T}{v^T \Delta f_k} \text{ for some } v_k \tag{6}$$

Usually, $v_k$ is taken as $\Delta f_k$ or $G_k^T \Delta x_k$:

- If $v_k = G_k^T \Delta x_k$, then $G_{k+1}$ minimizes $\|G_{k+1}^{-1} - G_k^{-1}\|_F$.
- If $v_k = \Delta f_k$, then $G_{k+1}$ minimizes $\|G_{k+1} - G_k\|_F$.

In both cases, the proof can be derived in straightforward manner from the proof of Theorem 4.1 in [10]. In general, it is not possible to give an estimation of the effect of the update of the preconditioner in terms of condition number. Nevertheless, it is possible to give a lower bound of condition number of the updated preconditioned linear system. Let $\{\sigma_k\}$ and $\{\tau_k\}$ be the singular values of $G_k J(x_{k+1})$ and $G_{k+1}J(x_{k+1}) = G_k J(x_{k+1}) + uw^T$ for $w^T = v^T J(x_{k+1})$. Then, the interlacing property of the singular values [13, Theorem 6.1] gives:

$$\begin{cases} \sigma_2 \leq \tau_1, \\ \sigma_{k+1} \leq \tau_k \leq \sigma_k,\ 1 < k < n \\ 0 \leq \tau_n \leq \sigma_{n-1}, \end{cases} \tag{7}$$

Then,

$$\kappa_2(G_{k+1}J(x_{k+1})) = \frac{\tau_1}{\tau_n} \geq \frac{\sigma_2}{\sigma_{n-1}}. \tag{8}$$

The same results can be derived for right preconditioned linear systems, since a rank-one update of the preconditioner linear system leads to a rank one modification of the preconditioned operator. This lower bound gives a limitation of the updating procedure: it will not be efficient if the preconditioned linear system has a large set of very high, or very low singular values.

Let us now illustrate the effect of the Broyden update on a manufactured problem. $-\Delta_{FD2}$ be the second order finite difference discrete 1D Laplacian operator for homogeneous boundary conditions, associated to the eigenpairs $\{(U_i,\lambda_i)\}_{1\leq i\leq n}$ such that $\lambda_i > \lambda_{i+1}$. We define the nonlinear function $F(v)$ vanishing for $v = 0$ and its Jacobian $J(v)$ definite positive matrix, of eigenpairs $\{(\eta_i U_1 + U_i, \mu_i)\}_{1\leq i\leq n}$, with condition number $\kappa_2(J(v)) = \frac{\mu_1}{\mu_n}$:

$$F(v) \stackrel{def}{=} \underbrace{(v,v)U_1U_1^T v}_{nonlinear} - \underbrace{\Delta_{FD2} v}_{linear} \tag{9}$$

$$J(v)h \;=\; 2(v,h)U_1U_1^T v + (v,v)U_1U_1^T h - \Delta_{FD2} h \tag{10}$$

$$\mu_1 = (2(v,U_1)^2 + (v,v) + \lambda_1), \mu_i = \lambda_i, 2 \leq i \leq n, \tag{11}$$

$$\eta_1 = 0, \eta_i = \frac{2(v,U_1)(v,U_i)}{\mu_i - \mu_1}, 2 \leq i \leq n. \tag{12}$$

For the sake of simplicity in calculus, starting from $X^0 = x_1^0 U_1$ and $G_0 = J(X^0)^{-1}$, Newton's and Broyden's iterates give the same $X^1 = \frac{2(x_1^0)^3}{\mu_1^0} U_1$ and the eigenvalue of $G_1 J(X^1)$ associated to $U_1$ is given by:

$$(G_1 J(X^1)U_1 = \frac{{\lambda_1}^3 + 6\,(x_1^0)^2 {\lambda_1}^2 + 9\,\lambda_1 (x_1^0)^4 + 12\,(x_1^0)^6}{{\lambda_1}^3 + 7\,(x_1^0)^2 {\lambda_1}^2 + 17\,\lambda_1 (x_1^0)^4 + 19\,(x_1^0)^6} U_1$$

These results suggest that $Z_1$ is a good preconditioner for $J(X_1)$ if $X^0$ is close to the solution $X = 0$, ($G_1 J(X^1)$ have the same $(n-1)$ eigenpairs $(U_i, 1), 2 \leq i \leq n$).

## 2 Application to the RAS Preconditioner

The Restrictive Additive Schwarz preconditioner of the linear system $J(x)\Delta x = -f(x)$ decomposed in $s$ overlapping subdomains, is given by:

$$M_{RAS}^{-1} = \sum_{i=1}^{s} \tilde{R}_i^T J^i(x)^{-1} R_i \tag{13}$$

where $R_i$ is the restriction operator of the $ith$ subdomain including the overlap, and $\tilde{R}_i$ is the restriction operator except that only interior nodes have a corresponding nonzero line. The matrix $J^i(x)$ is the submatrix of $J(x)$ corresponding to the $ith$ subdomain including the overlap. We propose to performing Broyden's updates starting from the RAS preconditioner $G_0 = M_{RAS}^{-1} = \sum_i \tilde{R}_i J^i(x)^{-1} R_i^T$.

Algorithm 1 gives an overview of the method for ($v_k = \Delta f_k$) within a time-stepper. Finding an optimal restarting criterion is out of the scope of this paper. One should notice that the restart may not happen at each time step. Hence, two simple strategies could be (1) to restart every $r$ time steps, or (2) to restart when a maximum number of Krylov iterations has been reached for the solution of the previous linear system.

**Algorithm 1** Time stepper with update of the RAS preconditioner

**Require:** restart parameter, initial guess $x$, $k = 0$
**for** each time step **do**
  *// Newton iterations:*
  **repeat**
    **if** restart **then**
      $G_0 \leftarrow \sum_i \tilde{R}_i^T J_i(x_0)^{-1} R_i$ *// Local LU factorizations*
      $k \leftarrow 0$
    **end if**
    solve $J(x)\Delta x = -f(x)$ with a Krylov method preconditioned by $G_k$.
    $x \leftarrow x + \Delta x$
    $G_{k+1} = G_k + (\Delta x_k - G_k \Delta f_k) \frac{f_k^T}{f_k^T \Delta f_k}$
    $k \leftarrow k+1$
  **until** convergence
**end for**

Therefore, even if $G_0$ is a sparse matrix, $G_k$ is not. Consequently, the matrix $G_k$ is never formed, we only compute its application to a vector. Let $u_k$ be $(\Delta x_k - G_k \Delta f_k)/(v^T \Delta f_k)$ then the application of $G_k$ to an arbitrary vector $x$ depends on the choice made for $v_k$ :

- For $v_k = \Delta f_k$ the application of the preconditioner can be rewritten as:

$$G_{k+1}x = G_0 x + \sum_{i=0}^{k} u_i v_i^T x = G_0 x + [u_0 \cdots u_k][v_0 \cdots v_k]^T x \tag{14}$$

  Hence, the additional cost of the application of $G_k$ compared to $G_0$ is roughly two matrix-vector products of $n \times k$ matrices. Furthermore, the computation of $u_k$ involves one application of $G_k$. One should also notice that the local LU factorizations can also be computed asynchronously, continuing Newton iterations during the computation of the restarted preconditioner.
- For $v_k = G_k^T \Delta x_k$, the explicit computation of $v_k$ should be avoided because it involves $G_k^T$, so $M_{RAS}^{-T}$ which cannot be easily computed. Then $G_{k+1}x$ is usually rewritten as in Eq. (15).

$$G_{k+1}x = \left(\prod_{i=k}^{0}(I - u_i \Delta x_i{}^T)\right) G_0 x \tag{15}$$

Following an idea of Martínez [16], Bergamaschi et al. proved in [3, theorem 3.6] that for $G_0$ and $x_0$ good enough initial guesses, the norm $\|I - G_k J(x_k)\|$ can be made arbitrarily small. Since the preconditioner is also reused from one time step to another, it slowly loses its efficiency and the algorithm must be restarted, which means recomputing $G_0$.

In terms of condition numbers, the preconditioner $G_k$ is not expected to be more efficient than the RAS preconditioner $M_{RAS}^{-1}$ of the current Jacobian matrix $J(x_k)$, but it's computational cost is less important: computing $G_{k+1}$ form $G_k$ does not involves LU factorizations unlike the computation of a new $M_{RAS}^{-1}$.

The efficiency of the updated preconditioner is expected to decrease from one time step to another, but this decrease should be slowed by the update. This decrease can be roughly explained by the fact that the convergence of Broyden's method is slower than the convergence of Newton's method. Thus, a restart of the algorithm is needed. This restart (Algo.1, step 4) consists in the computation of a new $G_0 = M_{RAS}^{-1}$ (i.e. new local LU factorizations).

One of the main drawbacks of the method presented here is the increase of the memory cost by two vectors per update. A few techniques can be used to reduce this memory cost: the simplest one consists in restarting the algorithm when a maximum number of updates is reached. One may also compress the updates using a truncated SVD of $[u_0 \cdots u_k][v_0 \cdots v_k]^T$ [18].

The parallelism of Equations (14) and (15) should also be discussed:

- The application of the preconditioner in Eq. (14) to a vector $x$ involves global communications since the matrices $[u_0 \cdots u_k]$ and $[v_0 \cdots v_k]$ are dense, and distributed over the processors. Then, depending on the implementation, Eq. (14) requires an additional global reduction of $k$ values, or $k$ reductions where the $k-1$ first are overlapped by computations.
- The parallel implementation of Eq. (15) requires $k$ sequential collective reductions. Hence, one should not use $v_k = G_k^T \Delta x_k$ for a parallel implementation on distributed memory computers.

## 3 Numerical experiments

Let us first give a numerical illustration of the model problem $F(v) = c$ where $F$ is from Eq. (9), $c \in \mathbb{R}^{100}$ an arbitrary vector, and starting from $G_0 = M_{RAS}^{-1}(-\Delta_{FD2})$. Then, the condition numbers are: $\kappa_2(J(X^1)) = 1.8 \times 10^9$, $\kappa_2(G_0 J(X^1)) = 1.7 \times 10^8$ and $\kappa_2(G_1 J(X^1)) = 1.2 \times 10^3$ when the preconditioner $G_0$ is updated with Broyden's update. This suggests that the update of the preconditioner has efficiently reduced the effect of the first eigenvalue of $J(X^1)$. We now consider the lid-driven cavity problem on the unit square. The PETSc library [1] was used for the implementa-

tion. In particular, the implementation of the following $(u,v,\omega,T)$-formulation is provided as a PETSc example [9]. The linear solver used in these experiments is a BiCGstab [17] and Jacobian matrices are approximated by a coloring method.

$$\begin{cases} -\Delta(u) - \nabla_y(\omega) = 0 \\ -\Delta(v) + \nabla_x(\omega) = 0 \\ \dot{\omega} - \Delta(\omega) + \nabla \cdot ([u \times \omega, v \times \omega]) - \nabla_x(T) = 0 \\ \dot{T} - \Delta(T) + \nabla \cdot ([u \times T, v \times T]) = 0 \end{cases} \tag{16}$$

Where $u$ and $v$ are the two components of the velocity field, $\omega = -\nabla_y u + \nabla_x v$ is the vorticity and $T$ the temperature. The space discretization is performed on a regular grid with a five-point stencil and the time discretization is a backward Euler scheme. The lid-velocity $u(x,0)$ is a nonzero constant, the other boundary conditions satisfy $u = v = 0$, $T = 0$ on the left wall, and $T = 1$ on the right wall, $\partial T/\partial y = 0$ on the top and the bottom. A fixed time step length has been used for the simulation, excepted for the very first time steps. The initial solution is zero everywhere excepted on the walls, and the solution at the previous time step is used as the initial guess for the current time step. In the following results, the linear systems are right preconditioned and $G_0$ is the RAS preconditioner of the current approximation of the Jacobian matrix, and the overlapping size is one. The reason is that when the left preconditioning technique is used, the natural stopping criterion of the Kyrlov method is based on the norm of the preconditioned residual. Hence, in order to compare two different preconditioners, one should use a stopping criterion based on the norm of true residual. The Newton iterations are stopped (i.e. the time step is accepted) when the absolute norm of the residual is lower than $10^{-6}$.

**Table 1** Comparison of the updated and the frozen preconditioner for a $512 \times 512$ grid decomposed in $8 \times 8$ subdomains. The lid velocity is $u(x,0) = 500$ and the time step length is $10^{-3}$. 1000 time steps are performed, and the sum of all the BiCGstab iterations is given. The algorithm is restarted every $f_r$ time steps, and the walltimes are given in seconds.

| | With update | | Without update | | Saved | Saved |
|---|---|---|---|---|---|---|
| $f_r$ | BiCGstab it. | Walltime | BiCGstab it. | Walltime | iterations (%) | walltime (%) |
| 1 | 34483 | 4729 | 34882 | 4744 | 1.144 | 0.309 |
| 5 | 34572 | 3820 | 35230 | 3850 | 1.868 | 0.779 |
| 40 | 35165 | 3609 | 35946 | 3649 | 2.173 | 1.085 |
| 60 | 35785 | 3619 | 36249 | 3625 | 1.280 | 0.159 |
| 80 | 36110 | 3653 | 36693 | 3670 | 1.589 | 0.461 |

Table 1 compares the total number of BiCGstab iterations with and without the rank-one update, for different frequencies of restarting. A frequency of restarting $f_r$ of 10 means that 100 local LU factorizations have been computed on each processors during the 1000 time steps. There is actually between one and three Newton iterations per time steps. This results show that the total number of Krylov iterations is slightly reduced by the updating method. If we take into account only the 580

time steps for which three Newton iterations have been performed, then 3.79% of the Krylov iterations have been saved. the additional cost of the application of the preconditioner is the reason why the proportion of Krylov iterations that are saved is greater than the proportion of saved computational time.

**Table 2** Number BiCGstab iterations for the updated and the frozen preconditioner. The grid decomposition is regular, using the same number of subdomains in each direction. 1000 time steps of length $10^{-3}$ are performed. The algorithm is restarted every 40 time steps.

| Processors | Grid size | Lid velocity | With update | Without update | Saved .it (%) |
|---|---|---|---|---|---|
| 8 | $128^2$ | 100 | 9009 | 9474 | 4.908 |
| 8 | $128^2$ | 300 | 16358 | 16748 | 2.329 |
| 16 | $128^2$ | 100 | 12724 | 13275 | 4.150 |
| 16 | $128^2$ | 300 | 17961 | 18345 | 2.093 |
| 16 | $256^2$ | 300 | 20011 | 20805 | 3.816 |
| 64 | $256^2$ | 300 | 28408 | 30114 | 5.665 |
| 64 | $256^2$ | 500 | 32599 | 32889 | 0.882 |

Table 2 compares the number of BiCGstab iterations for different sizes of grid and lid velocities. This results show that the Broyden update of the preconditioner may leads to a significant reduction of the number of Krylov iterations. For a restarting frequency of 40, the percentage of saved iterations generally decreases when the lid velocity is increased. This suggests that a more appropriate restarting algorithm should be designed in order to preserve the efficiency of the update. It should be noticed that the results presented above are obtained for a fixed time step length. The efficiency of the update is expected to change if an adaptive time stepping algorithm is used since the step length is present on the diagonal of the Jacobian matrix.

## 4 Conclusions

We presented a very simple procedure to update the RAS preconditioner without loss of parallelism. This update leads to a decrease of the number of Krylov iterations, especially for the time steps that requires the largest number of Newton iterations. However, further developments are needed to achieve an efficient method. This quasi-Newton update of the preconditioner should be used with a well-parametrized restarting procedure, since the efficiency of the preconditioner decreases from one iteration to another. A natural extension of this work is to use higher-rank updates, like the multisecant update [11]. Techniques such as partial updates, or relaxed updates should also be investigated since they are expected to significantly improve the numerical efficiency of the updated preconditioner.

**Acknowledgements** This work has been supported by the French National Agency of Research (project ANR-12-MONU-0012 H2MNO4). Laurent Berenguer held a doctoral fellowship (32116

Euros in 2012-2013) from the Région Rhône-Alpes. Authors also thank the Center for the Development of Parallel Scientific Computing (CDCSP) of the University of Lyon 1 for providing us with computing resources.

## References

1. Balay, S., Brown, J., , Buschelman, K., Eijkhout, V., Gropp, W.D., Kaushik, D., Knepley, M.G., McInnes, L.C., Smith, B.F., Zhang, H.: PETSc users manual. Tech. Rep. ANL-95/11 - Revision 3.3, Argonne National Laboratory (2012)
2. Bergamaschi, L., Bru, R., Martínez, A.: Low-rank update of preconditioners for the inexact Newton method with SPD Jacobian. Math. Comput. Modelling **54**(7-8), 1863–1873 (2011)
3. Bergamaschi, L., Bru, R., Martínez, A., Putti, M.: Quasi-Newton preconditioners for the inexact Newton method. Electron. Trans. Numer. Anal. **23**, 76–87 (electronic) (2006)
4. Broyden, C.G., Dennis Jr., J.E., Moré, J.J.: On the local and superlinear convergence of quasi-Newton methods. J. Inst. Math. Appl. **12**, 223–245 (1973)
5. Cai, X.C., Gropp, W.D., Keyes, D.E., D., T.M.: Newton-KrylovSchwarz methods in CFD. In: Proceedings of the International Workshop on Numerical Methods for the NavierStokes Equations,, pp. 17–30. Vieweg, Braunschweig (1995)
6. Cai, X.C., Keyes, D.E., Venkatakrishnan, V.: Newton-Krylov-Schwarz: An implicit solver for CFD. In: Proceedings of the Eighth International Conference on Domain Decomposition Methods, pp. 387–400. Wiley, New York (1997)
7. Cai, X.C., Keyes, D.E., Young, D.P.: A nonlinear additive Schwarz preconditioned inexact Newton method for shocked duct flows. In: Domain decomposition methods in science and engineering (Lyon, 2000), Theory Eng. Appl. Comput. Methods, pp. 345–352. Internat. Center Numer. Methods Eng. (CIMNE), Barcelona (2002)
8. Cai, X.C., Sarkis, M.: A restricted additive Schwarz preconditioner for general sparse linear systems. SIAM J. Sci. Comput. **21**(2), 792–797 (electronic) (1999)
9. Coffey, T.S., Kelley, C.T., Keyes, D.E.: Pseudotransient continuation and differential-algebraic equations. SIAM J. Sci. Comput. **25**(2), 553–569 (2003)
10. Dennis Jr., J.E., Moré, J.J.: Quasi-Newton methods, motivation and theory. SIAM Rev. **19**(1), 46–89 (1977)
11. Fang, H.r., Saad, Y.: Two classes of multisecant methods for nonlinear acceleration. Numer. Linear Algebra Appl. **16**(3), 197–221 (2009)
12. Gebremedhin, A.H., Manne, F., Pothen, A.: What color is your Jacobian? Graph coloring for computing derivatives. SIAM Rev. **47**(4), 629–705 (electronic) (2005)
13. Greif, C., Varah, J.M.: Minimizing the condition number for small rank modifications. SIAM J. Matrix Anal. Appl. **29**(1), 82–97 (electronic) (2006/07)
14. Keyes, D.E.: Aerodynamic applications of Newton-Krylov-Schwarz solvers. In: Fourteenth International Conference on Numerical Methods in Fluid Dynamics (Bangalore, 1994), *Lecture Notes in Phys.*, vol. 453, pp. 1–20. Springer, Berlin (1995)
15. Knoll, D.A., Keyes, D.E.: Jacobian-free Newton-Krylov methods: a survey of approaches and applications. J. Comput. Phys. **193**(2), 357–397 (2004)
16. Martínez, J.M.: An extension of the theory of secant preconditioners. J. Comput. Appl. Math. **60**(1-2), 115–125 (1995). Linear/nonlinear iterative methods and verification of solution (Matsuyama, 1993)
17. van der Vorst, H.A.: Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. SIAM J. Sci. Statist. Comput. **13**(2), 631–644 (1992)
18. Ziani, M., Guyomarc'h, F.: An autoadaptative limited memory Broyden's method to solve systems of nonlinear equations. Appl. Math. Comput. **205**(1), 202–211 (2008)