# Vary the s in Your s-step GMRES

David Imberti [*], Jocelyne Erhel [†]

December 4, 2017

**Abstract**

Krylov subspace methods are commonly used iterative methods for solving large sparse linear systems, however they suffer from communication bottlenecks on parallel computers. Therefore, $s$-step methods have been developed where the Krylov subspace is built block by block, so that $s$ matrix-vector multiplications can be done before orthonormalizing the block. Then Communication-Avoiding algorithms can be used for both kernels. This paper introduces a new variation on $s$-step GMRES in order to reduce the number of iterations necessary to ensure convergence, with a small overhead in the number of communications. Namely, we develop a $s$-step GMRES algorithm, where the block size is variable and increases gradually. Our numerical experiments show a good agreement with our analysis of condition numbers and demonstrate the efficiency of our variable $s$-step approach.

**Key words.** Communication-Avoiding, $s$-step Krylov subspace method, GMRES algorithm, variable $s$-step.

**AMS subject classifications.** 65F10, 65N22

## 1 Introduction

Many computational problems need to solve a large linear system $Ax = b$. Because linear solvers can be quite time consuming, they require an efficient implementation for supercomputers. An important class of methods is based on Krylov subspace methods like GMRES [34]. In this paper, we aim to improve the parallelization of such methods for general sparse matrices. Parallel GMRES algorithms have been studied by several authors, for example [8, 25, 5, 31, 1, 11, 14, 19, 28].

In Figure 1, we plot a tree, each branch of which describes a further subset of GMRES algorithms. In the course of this paper, we will briefly describe each branch, and the one we eventually take, by discussing each level of this tree from the root node down (a breadth-first traversal through GMRES).

Some parallelism can be found in the Arnoldi process of the usual restarted GMRES($m$) algorithm, where $m$ is the restarting parameter [4, 9, 13, 35]. However, global communications prevent good performance with many processors. Another way to build an orthonormal basis is to first compute a Krylov basis of size $m$, where $m$ matrix vector multiplications can be done in parallel, and to orthogonalize afterwards. We denote this algorithm as an $m$-step GMRES algorithm [1, 5, 8, 11, 14, 25, 28]. These two variants of restarted GMRES($m$), using either Arnoldi or a Krylov basis orthogonalized afterwards, are represented by the first level of the tree in Figure 1. However, if $m$ is large, the Krylov basis may become ill-conditioned [2, 19, 30]. Therefore, restarted $s$-steps methods have been defined, where the Krylov basis is built block by block, with $m/s$ blocks of size $s$, allowing $s$ parallel matrix vector multiplications and a better conditioned Krylov basis [19].

We propose a new variation of $m$-step and $s$-step GMRES algorithm, which avoids solving a triangular system. We denote the approach with the inverse of a triangular matrix as the 'Traditional' approach in

[*]INRIA, Campus universitaire de Beaulieu, 35042 Rennes Cedex, France
[†]INRIA, Campus universitaire de Beaulieu, 35042 Rennes Cedex, France

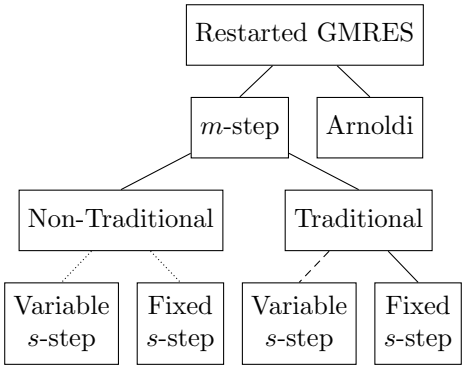


Figure 1: Variants of restarted GMRES(m).

Figure 1, while we propose a new 'Non-Traditional' branch. Both methods are mathematically equivalent and seem to have similar numerical behaviour, as illustrated in our numerical experiments. One key motivation to develop this 'Non-Traditional' branch is simplicity. From the 'Non-Traditional' fixed $s$-step version, we easily derive a new algorithm, where the block size $s$ is allowed to vary. It could be noted that a variable method could also be derived from the 'Traditional' branch, but we did not explore this potential branch for the sake of simplicity (therefore we use a dashed line in Figure 1). Indeed, it is much easier to vary the block size when there is no triangular system to solve. It can also be noted that variable $s$-step methods have been designed independently for other Krylov subspace methods, in particular Conjugate Gradient [6, 7].

The main novelty of this paper is thus the variable $s$-step GMRES algorithm, using a 'Non-Traditional' version of the subspace condition. We use dotted lines in Figure 1 to show the two algorithms described in this paper: 'Non-Traditional' fixed and variable $s$-step GMRES. Since convergence is related to the condition numbers of the blocks used to build the Krylov basis, we prove lower bounds for these condition numbers, which can be seen as the best possible case. Here we generalize the results for a symmetric matrix [2] to the case of a non-symmetric matrix with at least two different eigenvalues. In view of these bounds, we suggest to use an increasing block size, which could be adaptively defined. Since global communication occurs at each step, it is desirable to get a small number of steps, balancing the convergence rate and the communications overhead. We investigate a non adaptive sequence based of Fibonacci numbers, which results in a rapidly increasing block size and a number of steps of the same order as for a fixed block size, when the restarting parameter is large.

The paper is organized by discussing each level of the tree in Figure 1. In section 2, we give a brief background to restarted GMRES with Arnoldi and Traditional $m$-step GMRES (level 1). Then we introduce the Non-Traditional variant and compare both approaches for $m$-step GMRES (level 2). In section 3, we define the Non-Traditional fixed $s$-step GMRES algorithm and derive our variable $s$-step algorithm (level 3). Then we compare in section 4 the convergence and parallelism issues of both algorithms. Our numerical experiments in section 5 demonstrate the efficiency of a variable block size compared to a fixed block size. We observe a faster convergence, which is closely related to the condition numbers of the blocks. Finally, we conclude in section 6.

Throughout the paper, we use the Euclidean norm.

# 2 Traditional and Non-Traditional $m$-step GMRES

## 2.1 Restarted GMRES

We first recall the GMRES algorithm to build upon and recall $m$-step GMRES.

Let $Ax = b$ be a linear system, with $A$ a large sparse nonsymmetric nonsingular matrix of size $n$.

We introduce the Krylov subspace

$$\mathcal{K}_m = span\{r_0, Ar_0, A^2r_0, \cdots, A^{m-1}r_0\}$$

with the residual vector defined as $r_0 = b - Ax_0$ for some chosen initial vector $x_0$.

We also introduce $v_1 = r_0/\beta$, where $\beta = \|r_0\|$.

It is known that performing the Arnoldi process on $A$ and $r_0$ generates an orthonormal basis,

$$V_{m+1} = \{v_1, \ldots, v_{m+1}\}$$

of the Krylov subspace $\mathcal{K}_{m+1}$ and a upper Hessenberg matrix $\overline{H}_m$, of size $(m+1) \times m$, which satisfy the Arnoldi relation

$$AV_m = V_{m+1}\overline{H}_m. \tag{1}$$

The GMRES algorithm is based on the subspace condition $x_m \in x_0 + \mathcal{K}_m$, which can be written as

$$x_m = x_0 + V_m y.$$

Using the Arnoldi relation (1), we can say that $r_m = r_0 - AV_m y = V_{m+1}(\beta e_1 - \overline{H}_m y)$, where $e_1$ is the first column of the identity matrix.

The residual in GMRES satisfies the Galerkin condition $\min_{x \in x_0 + \mathcal{K}_m} \|b - Ax\|$, which is equivalent to the linear least squares problem

$$min_y \left\|\beta e_1 - \overline{H}_m y\right\|.$$

Restarted GMRES repeats this Arnoldi cycle for a new initial vector by setting $x_0 = x_m$ [20, 24, 27, 33, 34].

The GMRES algorithm may be modified to allow preconditioning, but this will not affect our discussion. Indeed, let us consider a preconditioned system $AM^{-1}(Mx) = b$, where $M$ is a nonsingular matrix. Then we can replace $A$ by $AM^{-1}$ everywhere and the subspace condition is written $x_m = x_0 + M^{-1}V_m y$. Thus, a matrix-vector product involves first solving a system with $M$ then multiplying by $A$. We will discuss how to parallelize this operation in section 4.1.

We will also assume throughout the paper that the Krylov subspace $\mathcal{K}_m$ is of dimension $m$, so that the residual of minimal norm is unique.

In summary, and to provide a comparison to variable $s$-step GMRES later, restarted GMRES, denoted by GMRES($m$), is expressed by Algorithm 1. It can be noted that it is not necessary to compute $x_k$ at each step since the norm of the residual can be estimated by $\|\beta e_1 - \overline{H}_k \overline{y}_k\|$. We leave it here for the sake of clarity.

**Algorithm 1** GMRES($m$)

1: **while** not converged **do**
2: $r_0 = b - Ax_0$
3: $\beta = \|r_0\|$
4: $v_1 = r_0/\beta$
5: $V_1 = \{v_1\}$
6: **for** k=1,m **do**
7: $w_k = Av_k$
8: Arnoldi process: orthogonalize $w_k$ against $V_k$ and normalize
9: Arnoldi relation: $AV_k = V_{k+1}\overline{H}_k$
10: solve the least squares problem $\overline{y}_k = \arg\min_y \|\beta e_1 - \overline{H}_k y\|$
11: compute $x_k = x_0 + V_k\overline{y}_k$
12: test convergence
13: **end for**
14: **if** not converged **then**
15: $x_0 = x_m$
16: **end if**
17: **end while**

## 2.2 Traditional $m$-step GMRES

The purpose of $m$-step GMRES is to improve the communication efficiency of Algorithm 1. Most of the computational time in Algorithm 1 is spent inside of the Arnoldi loop. Indeed, the least squares problem in step 9 of Algorithm 1 involves the small upper-Hessenberg matrix $\overline{H}_k$, therefore it may quickly be solved by a series of Givens rotations.

The Arnoldi process contains two key kernels: matrix-vector products and orthonormalization. To motivate $m$-step GMRES, we look at the problems involved in parallelizing these kernels.

The Arnoldi process builds an orthonormal basis of the Krylov subspace $\mathcal{K}_{m+1}$ with one matrix vector multiplication at a time and orthonormalizes it against the previous vectors as soon as it is added. The specific method for orthonormalization may vary. Regardless, all such methods lead to communication issues due to the global communication necessary in the dot product operation. A classical Gram-Schmidt orthonormalization reduces communication, compared to a modified Gram-Schmidt process, but then this procedure must be performed twice to ensure numerical stability [15]. Another possibility is to use Householder transformations, but this is more computationally intensive [38]. Parallelism in Gram-Schmidt process can be added by using a block Householder or QR method instead [21].

A Krylov basis can be built before orthonormalizing. This computation involves merely a succession of $m$ matrix-vector products (thus the name $m$-step GMRES), and is followed by an orthonormalization process. These two steps provide more parallelism by avoiding global communications, see subsection 4.2 for more details. However, convergence issues can arise with the choice of the Krylov basis, see subsection 4.1 for more details.

To detail the $m$-step GMRES procedure further we introduce a basis $W_{m+1}$ of the Krylov subspace $\mathcal{K}_{m+1}$.

We further assume a relation

$$AW_m = W_{m+1}T_{m+1}, \tag{2}$$

where $T_{m+1}$ is a $(m+1) \times m$ matrix. For instance, the matrix $T_{m+1}$ associated to the monomial basis has a diagonal of 1 below the main diagonal and 0 elsewhere.

We compute an orthonormal basis of $\mathcal{K}_{m+1}$ with a QR factorization:

$$W_{m+1} = V_{m+1} R_{m+1}. \tag{3}$$

The diagonal elements of the upper triangular matrix $R_{m+1}$ are forced to be real positive, so that the QR factorization is unique [16].

Then we get

$$\begin{array}{rcl} AW_m & = & V_{m+1} R_{m+1} T_{m+1}, \\ AV_m & = & V_{m+1} \tilde{H}_m, \end{array} \tag{4}$$

where $\tilde{H}_m = R_{m+1} T_{m+1} R_m^{-1}$.

Thus the subspace condition of GMRES can still be written $x_m = x_0 + V_m y$, and the Galerkin condition ends up with the least squares problem

$$min_y \left\| \beta e_1 - \tilde{H}_m y \right\|. \tag{5}$$

If the Krylov subspace $\mathcal{K}_{m+1}$ is of dimension $m+1$, then the residual $r_m$ is uniquely defined by the Galerkin condition, and we can conclude that $m$-step GMRES and GMRES($m$) are mathematically equivalent.

We summarize these steps in Algorithm 2, which we call $m$-step GMRES.

**Algorithm 2** Traditional $m$-step GMRES

**while** not converged **do**
  $r_0 = b - Ax_0$
  $\beta = \|r_0\|$
  $v_1 = r_0/\beta$
  $W_{m+1}$ basis of $\mathcal{K}_{m+1}$ such as $AW_m = W_{m+1} T_{m+1}$
  $QR$ factorization $W_{m+1} = V_{m+1} R_{m+1}$
  $\tilde{H}_m = R_{m+1} T_{m+1} R_m^{-1}$.
  solve the least squares problem $\tilde{y}_m = \arg\min_y \|\beta e_1 - \tilde{H}_m y\|$
  compute $x_m = x_0 + V_m \tilde{y}_m$
  test convergence
  **if** not converged **then**
    $x_0 = x_m$
  **end if**
**end while**

In practice, Algorithm 2 may overwrite $W$ with $V$ in order to save memory. It contains three main steps: the matrix-vector products in the computation of $W$, its QR factorization, and the multiplication by the $R_m^{-1}$ factor. We now turn to consider this $R_m^{-1}$ step.

## 2.3 Non-Traditional $m$-step GMRES

We design a new version of $m$-step GMRES method without this $R_m^{-1}$ factor, comparable with the treatment in [38]. We denote as 'Traditional' Algorithm 2 that uses this $R_m^{-1}$ factor, and we propose a 'Non-Traditional' algorithm without $R_m^{-1}$.

Let $W_m$ be a basis of the Krylov subspace $\mathcal{K}_m$ for which we do not assume relation (2).

The subspace condition can be equivalently written as

$$x_m = x_0 + W_m y. \tag{6}$$

We still compute an orthonormal basis of $\mathcal{K}_{m+1}$ with a QR factorization:

$$[v_1, AW_m] = V_{m+1}R_{m+1} = V_{m+1}[e_1, H_m], \tag{7}$$

so that we get $AW_m = V_{m+1}H_m$, where $H_m$ is the Hessenberg matrix obtained by removing the first column of $R_{m+1}$.

The Galerkin condition is then equivalent to solving the least squares problem

$$min_y \left\|\beta e_1 - H_m y\right\|. \tag{8}$$

Again, this algorithm is mathematically equivalent to GMRES($m$), since it satisfies the subspace and Galerkin conditions. Using a relation (2), this approach may also overwrite $W$ with $V$ in order to save memory. Indeed, $W$ can be replaced by $V$ during the QR factorization (7), and $W$ thus $x$ can be expressed with $V$ using (2), in the same way as before. We summarize this new version of $m$-step GMRES in Algorithm 3.

**Algorithm 3** Non-Traditional $m$-step GMRES

**while** not converged **do**
  $r_0 = b - Ax_0$
  $\beta = \|r_0\|$
  $v_1 = r_0/\beta$
  Build a basis $W_m$ of $\mathcal{K}_m$
  $QR$ factorization $[v_1, AW_m] = V_{m+1}R_{m+1}$
  $H_m = R_{m+1}$ without the first column.
  solve the least squares problem $y_m = \arg\min_y \|\beta e_1 - H_m y\|$
  compute $x_m = x_0 + W_m y_m$
  test convergence
  **if** not converged **then**
    $x_0 = x_m$
  **end if**
**end while**

We now compare Traditional and Non-Traditional approaches. On the one hand, there is an inversion step with $R_m^{-1}$, and on the other hand there is a multiplication step with $W_m$ (assuming that one uses an orthonormalization technique that is careful to preserve orthonormality [18]).

It seems that numerical instabilities in the algorithm might simply swap places from the inversion of $R_m$ to the matrix multiplication by $W_m$. In some sense, both methods inherit the conditioning of matrix $R_m$. Here, we do not argue *a priori* that one method is more stable than the other. Therefore, we base our choice between the two algorithms on the ease of implementation. It will be more apparent as we generalize to a $s$-step GMRES formulation below, that our Non-Traditional approach is easier to implement. Indeed, we will not need to discuss the details of calculating and determining the $R_m^{-1}$ blocks and the additional indexing challenges that occur as a result of that.

We now turn to such a $s$-step method, using both basis $W$ and $V$ in our Non-Traditional way.

# 3 Fixed and variable Non-Traditional $s$-step GMRES

## 3.1 Fixed $s$-step GMRES

In the $m$-step GMRES method, the size of the Krylov basis $W_m$ is equal to the restarting parameter $m$. However, the condition number of $AW_m$ and thus of $H_m$ increases with $m$, limiting the restarting parameter to small values. But quite often convergence could stall if the restarting parameter is too small [28].

Therefore, another approach can be taken where the Krylov orthonormal basis $V_{m+1}$ is built by successively computing blocks $B_j$ of size $s \leq m$ [19]. Each restarting cycle will then be composed of several steps $j$ where a block $B_j$ is added to the basis $W$ and $AB_j$ is orthonormalized, adding a new block to the orthonormal basis $V$. This should allow taking $s$ sufficiently small for limiting the condition numbers and $m$ sufficiently large for avoiding stagnation. If $s$ is large enough, parallelism can occur by decoupling matrix-vector multiplications from orthonormalization. The optimal value of $s$ depends on the linear system considered and on the computer architecture.

For the sake of simplicity, we assume that $m$ is a multiple of $s$ so that $m = sJ$. We use the Non-Traditional way as described above, as opposed to the Traditional approach characterized by [19].

Let $B_1$ be a basis of the Krylov subspace $\mathcal{K}_s$ and let $W_s = B_1$. We perform a QR factorization as in equation (4):

$$[v_1, AW_s] = V_{s+1}R_{s+1}. \tag{9}$$

The last vector $v_{s+1}$ of the orthonormal system $V_{s+1}$ will serve as the initial vector of the next block $B_2$. We may inductively give a more general definition of $W$, the orthonormal system $V$, and the triangular matrix $R$. At step $j$, with $2 \leq j \leq J$, we assume that we have built $W_{s(j-1)}$ of size $s(j-1)$ and $V_{s(j-1)+1}$ of size $s(j-1)+1$, with the last vector $u = v_{s(j-1)+1}$. The block $B_j$ of size $s$ is then a basis of the Krylov subspace $\mathcal{K}_s(u)$, where $\mathcal{K}_s(u) = span\{u, Au, A^2u, \cdots, A^{s-1}u\}$.

We can now describe a recursive relation for $W_{sj}$:

$$W_{sj} = [W_{s(j-1)}, B_j] = [B_1, B_2, \ldots, B_j]. \tag{10}$$

Now, we show by induction how to perform a QR factorization of $[v_1, AW_{sj}]$. Let us assume that

$$[v_1, AW_{s(j-1)}] = V_{s(j-1)+1}R_{s(j-1)+1},$$

which is true for $j = 2$. We orthogonalize the vectors $AB_j$ against the basis $V_{s(j-1)+1}$ to get new $s$ vectors and $s$ columns:

$$AB_j = V_{sj+1}\begin{pmatrix}\overline{S}_j \\ S_j\end{pmatrix},$$

where $S_j$ is an upper triangular matrix of size $s$. We get

$$[v_1, AW_{sj}] = V_{sj+1}R_{sj+1}, \tag{11}$$

where

$$R_{sj+1} = \begin{pmatrix}R_{s(j-1)+1} & \overline{S}_j \\ 0 & S_j\end{pmatrix}.$$

We define the upper Hessenberg matrix $H_{sj}$ as $R_{sj+1}$ without the first column $e_1$ so that

$$AW_{sj} = V_{sj+1}H_{sj}. \tag{12}$$

Before deriving the $s$-step method, we prove that the systems $W_{sj}$ and $V_{sj}$ span Krylov subspaces.

**Theorem 1.** *We assume that the dimension of $\mathcal{K}_{m+1}$ is equal to $m+1$. Let $W_{sj}$ defined by (10) and $V_{sj+1}$ defined by (11). Then $W_{sj}$ is a basis of $\mathcal{K}_{sj}$ and $V_{sj+1}$ is an orthonormal basis of $\mathcal{K}_{sj+1}$.*

*Proof.* The proof is by induction.

For $j = 1$, $W_s$ is a basis of $\mathcal{K}_s$, also $[v_1, AW_s]$ is a basis of $\mathcal{K}_{s+1}$ and by (9), $V_{s+1}$ is an orthonormal basis of $\mathcal{K}_{s+1}$.

Now, we assume that $W_{s(j-1)}$ is a basis of $\mathcal{K}_{s(j-1)}$ and that $V_{s(j-1)+1}$ is a basis of $\mathcal{K}_{s(j-1)+1}$.

Thus $u = v_{s(j-1)+1} \in \mathcal{K}_{s(j-1)+1} \setminus \mathcal{K}_{s(j-1)}$ and by construction $B_j \in \mathcal{K}_{sj}$. Thanks to relation (10) and to the induction assumption, we conclude that $W_{sj} \in \mathcal{K}_{sj}$.

Moreover, $u \notin \mathcal{K}_{s(j-1)}$ thus the block $B_j$ is linearly independent from the system $W_{s(j-1)}$ and $W_{sj}$ is a basis of $\mathcal{K}_{sj}$ which is of dimension $sj$. Also $[v_1, AW_{sj}]$ is a basis of $\mathcal{K}_{sj+1}$ and by (11), $V_{sj+1}$ is an orthonormal basis of $\mathcal{K}_{sj+1}$. This completes the inductive step.

□

Therefore, the subspace condition at each step $j$ can be written for the Krylov subspace $\mathcal{K}_{sj}$ as

$$x = x_0 + W_{sj} y. \tag{13}$$

And, with the Hessenberg matrix $H_{sj}$ from equation (11), the Galerkin condition is again a least squares problem

$$min_y \left\| \beta e_1 - H_{sj} y \right\|. \tag{14}$$

We summarize this fixed $s$-step GMRES, which we call SGMRES($m$,$s$), in Algorithm 4.

**Algorithm 4** Non-Traditional Fixed $s$-step SGMRES($m$,$s$)

**while** not converged **do**
  $r_0 = b - Ax_0$
  $\beta = \|r_0\|$
  $v_1 = r_0/\beta$
  $W_0 = \emptyset$ and $V_1 = [v_1]$
  $J = m/s$
  **for** $j = 1, J$ **do**
    $u = v_{s(j-1)+1}$
    $B_j$ a basis of $\mathcal{K}_s(u)$
    $W_{sj} = [W_{s(j-1)}, B_j]$
    orthonormalize $AB_j$ against $V_{s(j-1)+1}$ to obtain the last $s$ vectors $[v_{s(j-1)+2}, \ldots, v_{sj+1}]$ and the last $s$ columns of $R_{sj+1}$
    $H_{sj} = R_{sj+1}$ without the first column
    solve the least squares problem $y_F = \arg\min_y \|\beta e_1 - H_{sj} y\|$
    compute $x_{sj} = x_0 + W_{sj} y_F$
    test convergence
  **end for**
  **if** not converged **then**
    $x_0 = x_m$
  **end if**
**end while**

One should note that if $s = 1$, one obtains restarted GMRES (Algorithm 1), and if $s = m$, one obtains $m$-step GMRES (Algorithm 3). Thus, Algorithm 4 represents a generalization of GMRES algorithms.

It is noteworthy to recall that a Traditional fixed $s$-step approach is possible [19], but requires intensive care in the appropriate block indices in order to generate and treat the $R^{-1}$ factors.

## 3.2 Variable $s$-step GMRES

The fixed $s$-step GMRES method aims at improving the condition number of the basis $W$ compared with $m$-step GMRES, at the price of less parallelism. Nevertheless, if $s$ is chosen too large, $W$ could be ill-conditioned and convergence might stagnate, as illustrated in the numerical experiments of section 5.

To help balance these two concerns, we propose a new method, where the block size $s$ is not fixed, but variable. It is not easy to choose an optimal value of $s$ between 1 and $m$. The idea behind a variable approach is to choose adaptively the block size to cope with conditioning and parallelism issues, as was done for deflation in [28] and for CG in [7].

In this variable approach, at each step $j$, we define a new block size $s_j$, and build a system $W_{l_j}$ of size $l_j$, where $l_0 = 0$ and $l_j = l_{j-1} + s_j$, $j \geq 1$. We assume that $m = l_J$ for some integer $J$.

In Algorithm SGMRES($m$,$s$), we simply have $s_j = s$ and $l_j = sj$, whereas the variable block size $s_j$ could be chosen adaptively, by using some criterion of convergence.

Because we do take a Non-Traditional approach, we may easily describe this variable approach. As such, all we need to do is to replace $sj$ with $l_j$ appropriately above in Algorithm 4 in order to properly describe a variable $s$-step method. We detail this more formally below.

As before, we give a recursive definition of $W$, the orthonormal system $V$ and the triangular matrix $R$. Let $W_0 = \emptyset$ and $V_1 = [v_1]$.

At step $j$, with $1 \leq j \leq J$, we assume the existence of $W_{l_{j-1}}, V_{l_{j-1}+1}$ and we define the block $B_j$ as a basis of $\mathcal{K}_{s_j}(u)$, where $u = v_{l_{j-1}+1}$ is the last vector of the orthonormal system $V_{l_{j-1}+1}$.

We define the system $W_{l_j}$ recursively:

$$W_{l_j} = [W_{l_{j-1}}, B_j] = [B_1, B_2, \ldots, B_j], \tag{15}$$

then perform a QR factorization by orthogonalizing the last block $AB_j$, and get:

$$[v_1, AW_{l_j}] = V_{l_j+1} R_{l_j+1}. \tag{16}$$

Introducing the upper Hessenberg matrix $H_{l_j}$ as $R_{l_j+1}$ minus the first column $e_1$, we get

$$AW_{l_j} = V_{l_j+1} H_{l_j}. \tag{17}$$

We can reiterate the proof of Theorem 1 to get a very similar Theorem.

**Theorem 2.** *We assume that the dimension of $\mathcal{K}_{m+1}$ is equal to $m+1$. Let $W_{l_j}$ defined by (15) and $V_{l_j+1}$ defined by (16). Then $W_{l_j}$ is a basis of $\mathcal{K}_{l_j}$ and $V_{l_j+1}$ is a basis of $\mathcal{K}_{l_j+1}$.*

Therefore, the subspace condition at each step $j$ can be written for the Krylov subspace $\mathcal{K}_{l_j}$ as

$$x = x_0 + W_{l_j} y, \tag{18}$$

and the Galerkin condition is again a least squares problem

$$min_y \left\| \beta e_1 - H_{l_j} y \right\|. \tag{19}$$

We summarize this variable $s$-step GMRES, which we call VGMRES(m,s), in Algorithm 5. The parameter $s$ is the maximal block size.

**Algorithm 5** Non-Traditional Variable $s$-step VGMRES($m$,$s$)

**while** not converged **do**
  $r_0 = b - Ax_0$
  $\beta = \|r_0\|$
  $v_1 = r_0/\beta$
  $W_0 = \emptyset$ and $V_1 = [v_1]$
  $l_0 = 0$
  **for** $j = 1, \ldots$ UNTIL $l_j \geq m$ **do**
    choose $s_j \leq s$
    $l_j = l_{j-1} + s_j$
    $u = v_{l_{j-1}+1}$
    $B_j$ a basis of $\mathcal{K}_{s_j}(u)$
    $W_{l_j} = [W_{l_{j-1}}, B_j]$
    orthonormalize $AB_j$ against $V_{l_{j-1}+1}$ to obtain $[v_{l_{j-1}+2}, \ldots, v_{l_j+1}]$ and the last columns of $R_{l_j+1}$
    $H_{l_j} = R_{l_j+1}$ without the first column
    solve the least squares problem $y_V = \arg\min_y \|\beta e_1 - H_{l_j} y\|$
    compute $x_{l_j} = x_0 + W_{l_j} y_V$
    test convergence
  **end for**
  **if** not converged **then**
    $x_0 = x_{l_j}$
  **end if**
**end while**

Algorithm 5 is thus a generalization of Algorithm 4, where $s_j = s$.

# 4 Convergence and communication issues

## 4.1 Analysis of condition numbers

In both Algorithms 4 and 5, numerical behaviour is directly related to the condition number $\kappa(H)$ to solve the least squares problem, and to the condition number $\kappa(W)$ to compute the approximate solution [18]. Since $AW = VH$, we get $\kappa(H) = \kappa(AW)$. Although algorithms are equivalent in exact arithmetic, they can behave differently in finite precision arithmetic. We observe and assume that large condition numbers $\kappa(H_{l_j})$ can slow down the convergence rate.

When the matrix is symmetric, the condition number of a monomial basis $B_j$ has an exponential growth with the block size $s$ [2]. Other bases can be used to reduce the condition number ([30] and references herein), such as a Newton basis [1, 14, 28], or a Chebyshev basis [22, 23].

Here, we consider a nonsingular matrix $A$, with complex eigenvalues such that $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n| > 0$. We get lower bounds of condition numbers which, in some sense, indicate the best case we can expect. Although we do not provide upper bounds, this result can highlight potential loss of convergence. We start by an easy but useful result.

**Lemma 1.** *Let* $W = [B_1, \cdots, B_j]$, *then* $\kappa(W) \geq \max_{1 \leq i \leq j} \kappa(B_i)$

*Proof.* Note that the singular values $\sigma_k(W)$ are the square roots of the eigenvalues $\lambda_k(W^T W)$ and that $\kappa(W) = \sigma_1(W)/\sigma_n(W)$.

Since $B_i^T B_i, 1 \leq i \leq j$, is a principal submatrix of $W^T W$, by Cauchy interlacing theorem, we get for all $i$ [39]

$$\begin{aligned} \sigma_1(W)^2 &= \lambda_1(W^T W) \geq \lambda_1(B_i^T B_i) = \sigma_1(B_i)^2, \\ \sigma_n(W)^2 &= \lambda_n(W^T W) \leq \lambda_n(B_i^T B_i) = \sigma_n(B_i)^2. \end{aligned} \tag{20}$$

Thus $\forall i,\ 1 \leq i \leq j,\ \kappa(W) \geq \kappa(B_i)$ and after taking the maximum on $i$ we have the result wanted.

□

Now we consider a block containing two vectors $A^{k-1}u, A^k u$ for some vector $u$ and some integer $k$. A monomial basis $B_j$ typically satisfies this property. The following theorem is comparable to similar Krylov subspace results [3]. In the same spirit as the power method [16], the two vectors $A^k u$ and $A^{k-1}u$ tend to be in the direction of the dominant eigenvector of $A$ for a large $k$. We analyse their impact on the condition number of a block which contains them.

**Theorem 3.** *Let $\lambda_i$ be the complex eigenvalues of a nonsingular matrix $A \in \mathbb{R}^{n\times n}$, such that $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n| > 0$. We assume that $A$ is diagonalizable. Let*

$$D = \left\{D_1, A^{k-1}u, A^k u\right\} \tag{21}$$

*for some integer $k$, some block $D_1$ with any given number of vectors, and some vector $u$, such that $Au \neq \lambda_1 u$.*

*Then there exists a constant $c$ which depends on the matrix $A$ and the vector $u$ such that*

$$\kappa(D) \geq c \left|\frac{\lambda_1}{\lambda_2}\right|^{k-1}$$

*Proof.* Let

$$E = \left\{0, 0, \lambda_1 A^{k-1}u - A^k u\right\}, \tag{22}$$

then $E \neq 0$ and $D + E$ is singular, so that

$$\kappa(D) \geq \frac{\|D\|}{\|E\|}.$$

First, we note that $\|D\| \geq \left\|A^k u\right\|$.

We consider the complex Schur factorization $A = QTQ^*$ with $Q$ unitary and $T$ upper triangular with the eigenvalues of $A$ on the diagonal. We assume that the eigenvalues are ordered such that $\lambda_1$ is the last entry in the diagonal of $T$.

Let $\delta = Q^* u$ so that $u = Q\delta$ and $Au = QT\delta$. Then $A^k u = QT^k\delta$ and $\left\|A^k u\right\| = \left\|T^k\delta\right\|$. Since $T$ is upper triangular with $\lambda_1$ in the bottom-right corner, we get $(T^k\delta)_n = \lambda_1^k\delta_n$, thus

$$\|D\| \geq \left|\lambda_1^k \delta_n\right|.$$

Second, $\|E\| = \left\|\lambda_1 A^{k-1}u - A^k u\right\|$. Let $X = (x_i)$ be a basis of eigenvectors of $A$, such that $A = X\Delta X^{-1}$, where $\Delta$ is the diagonal matrix of the eigenvalues. Let $\alpha = X^{-1}u$, so that $\|\alpha\| \leq \left\|X^{-1}\right\| \|u\|$. Then

$$\lambda_1 A^{k-1}u - A^k u = X\Delta^{k-1}(\lambda_1\alpha - \Delta\alpha)$$

Therefore

$$\|E\| \leq 2\left|\lambda_1\right|\left|\lambda_2\right|^{k-1}\kappa(X)\|u\|,$$

where $\kappa(X) = \|X\|\left\|X^{-1}\right\|$.

Finally

$$\kappa(D) \geq c\left|\frac{\lambda_1}{\lambda_2}\right|^{k-1},$$

where

$$c = \frac{|\delta_n|}{2\kappa(X)\,\|u\|}.$$

□

If we choose a monomial basis for each block $B_j$, then we get a lower bound for the condition number of $H$.

**Corollary 1.** *Let $W_{l_j}$ be the Krylov basis defined by a variable s-step method, where each block $B_i$ is a monomial basis of $\mathcal{K}_{s_i}(u)$. Under assumptions of theorem 3, there exists constants $c_i$ such that*

$$\kappa(H_{l_j}) \geq \max_{1 \leq i \leq j} c_i \left|\frac{\lambda_1}{\lambda_2}\right|^{s_i - 1}.$$

*Proof.* Since $AW_{l_j} = [AB_1, AB_2, \ldots, AB_j]$, and $AB_i = [Au, \ldots, A^{s_i-1}u, A^{s_i}u]$, where $u$ depends on $i$, we can apply Theorems 1 and 3, with $k = s_i$. □

These results show that, in the case of a fixed $s$-step method with a monomial basis,

$$\forall j \geq 1,\ \kappa(H_{sj}) \geq c \left|\frac{\lambda_1}{\lambda_2}\right|^{s-1}.$$

In the case of a variable $s$-step method with a monomial basis, if the first block size is $s_1 = s$, then we get the same result. We recall that $s_j$ should be as large as possible to avoid communication. Therefore, we advocate the use of a variable sequence with an increasing block size. Then the lower bound in Corollary 1 will increase gradually with $j$, and hopefully the condition number will behave similarly. The parameter $s_j$ could be chosen adaptively by estimating the condition number of $B_j$, with the constraint $s_{j-1} \leq s_j \leq s$. In the numerical experiments of this paper, we do not introduce this adaptivity but choose an increasing sequence *a priori*. The choice is based mainly on communication issues.

## 4.2 Communication Analysis

One last question remains. While we have studied the condition numbers in Algorithm 5, we have yet to show that the parallel costs are not prohibitive.

The two main kernels in a fixed or variable s-step method are the sequence of matrix-vector multiplications to compute the block $B_j$, and the orthonormalization of $AB_j$ to compute the new basis vectors of $V$ and the new columns of $H$. The first operation can be done efficiently in parallel by using parallelism in each matrix-vector multiplication, including a parallel preconditioning step based on domain decomposition [28]. When sub-domains are allocated to different processes, communications occur only between neighbouring processes, avoiding global communications. Without preconditioning, the block $B_j$ of a monomial basis is efficiently computed thanks to a matrix power kernel, avoiding also global communication [26]. Preconditioning based on incomplete factorization has been recently included in such a kernel [17], but it is still an issue to deal with general preconditioning and matrix power kernel. The orthogonalization of $AB_j$ can be done by using various parallel algorithms [36] such as RODDEC [28] or CA-QR [12].

These communication-avoiding kernels significantly improve the parallel performance. Nevertheless, solving the least-squares problem and checking convergence requires a global communication at each step of a restarting cycle. We are thus interested in reducing the number of steps in a restarting cycle, which is $m/s$ in a fixed s-step method and some $J$ in a variable s-step method, such that $l_J = m$. We assume that the variable sequence $s_j$ is increasing and is capped at $s$. Let $J_1$ be the step where the variable sequence is capped and $J_2$ the number of remaining steps after this cap, so that $J = J_1 + J_2$. Then $m = l_J = l_{J_1} + sJ_2$ and

$$J = J_1 + (m - l_{J_1})/s. \tag{23}$$

The objective is to use a small number $J_1$ with $l_{J_1} << m$, so that the number of steps $J$ is of the same order as $m/s$ and that most of the steps deal with a block size $s$.

### 4.3 Variable Fibonacci sequence

We now introduce a variable block size based on a Fibonacci sequence. We choose this sequence because it increases very fast and fulfills the objective above. Any other increasing sequence could be used, provided that most of the blocks are of size $s$.

| j | 1 | 2 | 3 | 4 | $J_1$ | $J_1+1$ | ... | $J_1+J_2=J$ |
|---|---|---|---|---|---|---|---|---|
| $s_j$ | 1 | 2 | 3 | 5 | $\leq s$ | s | ... | $s$ |
| $l_j$ | 1 | 3 | 6 | 11 | $l_{J_1}$ | $l_{J_1}+s$ | ... | $m$ |

Table 1: Capped Fibonacci s-step sequence.

We denote by FibGMRES(m,s) the special case of Algorithm 5 where the sequence of block sizes $s_j$ is given by Fibonacci numbers until the block size is capped at $s$, as shown in Table 1. Thanks to the properties of Fibonacci numbers, we can estimate $J_1$.

**Theorem 4.** *Let $J_1$ be the step where the variable Fibonnaci sequence is capped. Then, if $s$ is small compared to $m$, $J_1 = O(log_\phi(s))$, where $\phi = \frac{1+\sqrt{5}}{2}$.*

*Proof.* Assuming that we start here with $s_1 = 1, s_2 = 2$ as in Table 1, the block size $s_j$ is the $(j+1)$th Fibonacci number. Using the properties of the Fibonacci sequence [32], we get $l_{J_1} = fib(J_1+2)-2$, where $fib(J_1+2)$ is the $(J_1+2)$ Fibonnaci number.

For $j \leq J_1$, we have $s_j = \frac{\phi^{j+1}-\phi^{-(j+1)}}{\sqrt{5}}$ and $s_j \leq s$, so that

$$\begin{array}{rcl} \phi^{J_1+1} & = & \sqrt{5}s_{J_1} + \phi^{-(J_1+1)} \\ \phi^{J_1+1} & \leq & \sqrt{5}(s+1) \\ J_1 & \leq & log_\phi(\sqrt{5}(s+1)) - 1 \\ J_1 & = & O(log_\phi(s)) \end{array} \tag{24}$$

□

Therefore, the number of steps, thus of global communications, is of the same order $O(m/s)$, regardless of using SGMRES or FibGMRES, due to the rapid growth of the $s_j$ block size in FibGMRES. It should also be noted here the important playoff between convergence and communication issues. If $s$ is kept too small, then this increases $m/s$, thus the number of global communications. However, in the previous section we showed that $s$ being too large introduces a loss of convergence. Also, performance could get worse for very large values of $s$. This implies that there is a balance that must be struck between number of iterations and number of communications.

We now show some numerical experiments confirming this interplay, as well as the convergence properties of FibGMRES.

## 5 Numerical Results

We run numerical experiments with various matrices, with a fixed or variable block size. In all our tests, the block $B_j$ is a monomial basis of the Krylov subspace $K_{l_j}(u)$. We are aware that other choices would lead to better conditioned Krylov bases, but we reckon that these experiments highlight the impact of the block size and the differences between a fixed and a variable block size.

In the first part, we will regard the number of iterations of Traditional and Non-Traditional Algorithms 2 and 3 respectively. Namely, in our tests, the difference in convergence of these two methods is marginal, so that we prefer the simplicity of implementation of Algorithm 3.

In the second part, we will analyse the impact of the block size $s$ on convergence rate in the fixed s-step method. We compare SGMRES($m$,$s$) with $1 < s \leq m$ to GMRES($m$). We observe that the condition number of $AW_{sj}$ does not depend on $j$ but increases with $s$ and affects convergence, as we expected from Corollary 1.

After this we will focus on the heart of this paper, which is the convergence rate of variable $s$-step GMRES methods, in particular the Fibonacci $s$-step FibGMRES($m$,$s$) method, where the variable block size $s_j$ follows a Fibonacci pattern. We compare the three algorithms FibGMRES($m$,$s$), SGMRES($m$,$s$) and GMRES($m$), by looking at the convergence rate and at the condition number $\kappa(AW_{l_j})$. Our tests corroborate the result of Corollary 1. We run experiments with a block size up to 32, in order to measure convergence and condition numbers. In practice, such a large value would probably require to replace the monomial basis by another one, as in [27].

We developed some code in Octave and performed all experiments on a computer with a x86 architecture. In all our tests, the initial guess $x_0$ as well as the right-hand side $b$ are vectors whose components are uniform random numbers between 0 and 1. The relative residual at iteration $k$ is defined by $\|r_k\| / \|r_0\|$. We do not use a convergence test, but run two or three restarting cycles.

## 5.1 Experiments with $m$-step GMRES

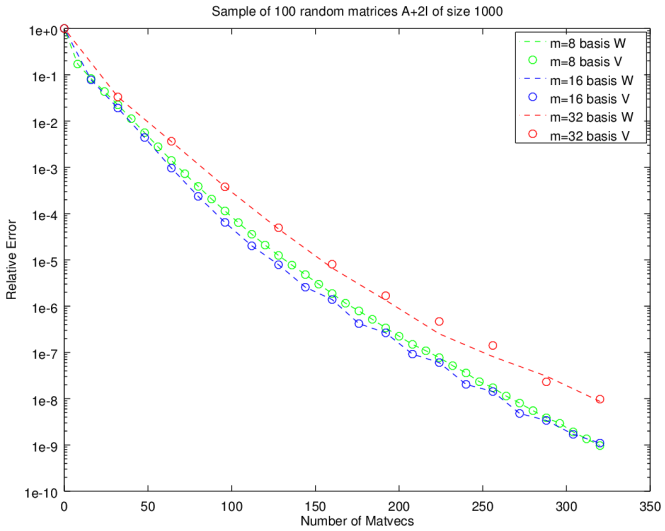


Figure 2: Comparison of Traditional and Non-Traditional variants of $m$-step GMRES(m). Mean error curves with a sample of 100 random matrices, using $m = 8, 16, 32$.

The first question is whether the use of $V$ or $W$ as a Krylov basis impacts the convergence of $m$-step methods. To get some insight into this question, we rely on a series of tests, done with samples of sparse matrices of type $A + 2I$, where the matrix $A$ is random and $I$ is the identity matrix. These matrices are of size $n = 1000$ with 10000 nonzeros. They are scaled and preconditioned by Jacobi.

We compare the two versions of m-step GMRES, with a block size equal to the restarting parameter,

where the basis is either $V$ or $W$. In Figure 2, we plot the mean absolute error $\left\|x_m - (A+2I)^{-1}b\right\|$ after each restarting cycle, for three values of $m$. The errors of the two versions, using either $V$ (Algorithm 2) or $W$ (Algorithm 3), are very similar. Instabilities seem to occur either in the inversion of $R_m$ or in the matrix multiplication by $W_m$. For these 100 matrices, our comparison ends up with a similar numerical behaviour. Therefore, we choose the basis $W$ for the sake of simplicity. From now on, we use Algorithms 4 and 5 in all our experiments.

It should be noted that with the addition of better preconditioners or with a well-conditioned matrix, what would change is essentially the rate of convergence. A similar situation would occur with the addition of a Newton basis, which could improve the condition numbers of $W_m$ and $R_m$. With classical restarted GMRES($m$), convergence is quite often faster by increasing the restarting parameter $m$ [34]. Thus, on one hand, a large value of $m$ is sometimes necessary to ensure convergence of restarted GMRES and on the other hand a small value of $m$ is often required to ensure a well-conditioned Krylov basis. The idea behind $s$-step methods with the block size $s$ smaller than the restarting parameter $m$, is to find a trade-off between convergence and parallelism issues. It can be noted that convergence can also be improved through deflation, see [28] for example.

## 5.2 Experiments with fixed $s$-step SGMRES($m$,$s$)

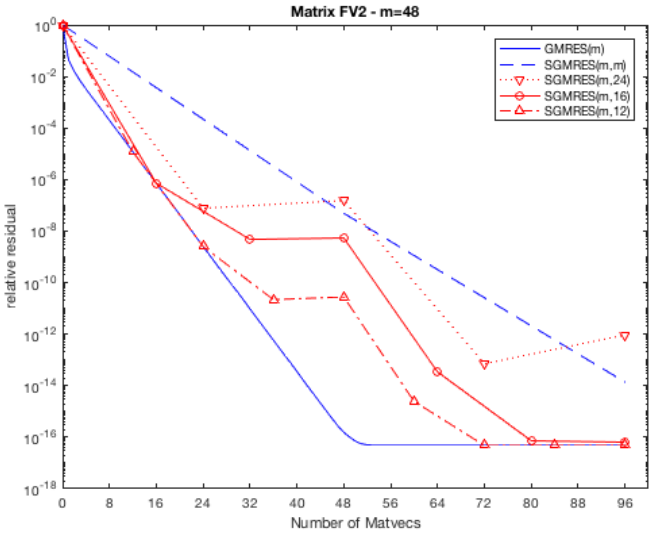


Figure 3: Impact of block size $s$ in SGMRES($m$,$s$). Convergence curves with the matrix fv2, using $m = 48$ and $s = 12, 16, 24, 48$.

Here, we analyze the convergence and the condition numbers of the fixed s-step method, denoted by SGMRES($m$,$s$).

We use the matrix fv2 from the University of Florida matrix collection, which is of size $n = 9801$ with $nz = 87025$ nonzero elements [10]. This matrix is symmetric and well-conditioned so that restarted GMRES($m$) converges quickly, without preconditioning and with a relatively small value of $m$.

We first analyze the impact of the block size $s$ on convergence and condition numbers, for a given restarting parameter $m = 48$.

Figure 3 represents the relative residual during two cycles. The primary feature to notice is the bounding behaviour by standard GMRES($m$) and SGMRES($m, m$), which correspond respectively to the

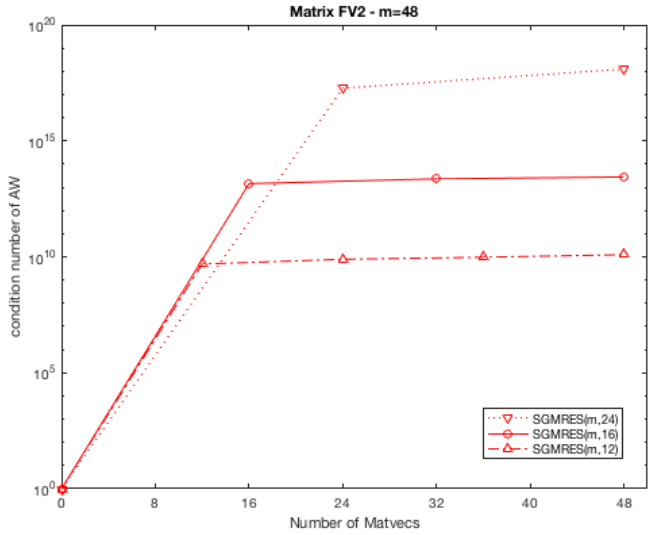


Figure 4: Impact of block size $s$ in SGMRES($m$,$s$). Condition number of $AW_{sj}$ with the matrix fv2, using $m = 48$ and $s = 12, 16, 24$.

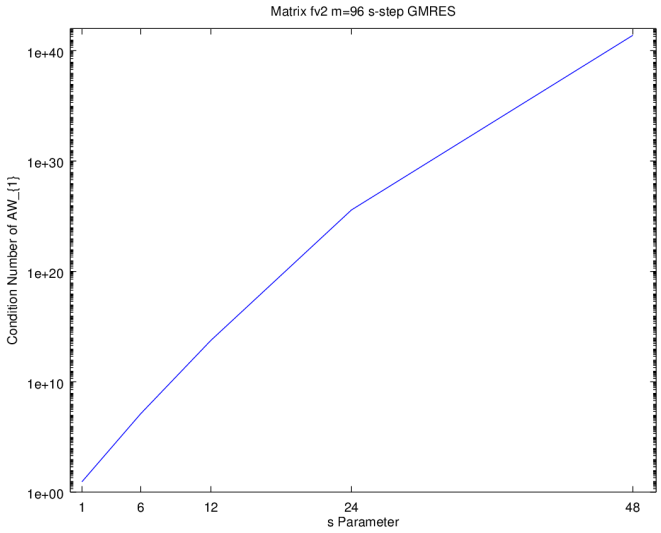


Figure 5: Condition number of $AW_s$ as a function of $s$. Results with the matrix fv2, using $m = 96$.

bounding values $s = 1$ and $s = m$ of the block size. As expected, GMRES($m$) is the best case, whereas SGMRES($m, m$) is the worst case in general.

A 'hockey-stick' pattern occurs for all $s$ values during each restarting cycle, where just before restarting the residual norm has a mild plateau in the case of $s = 12$, but can even begin to increase slightly for $s = 16$ and $s = 24$. In this last case, the residual becomes even larger than for $s = m$. Meanwhile, this behaviour is well corroborated by Figure 4, which shows the condition number of $AW_{sj}$ during the first

cycle. This one is quite large already for the first block $AW_s$ and then plateaus for other blocks $AW_{sj}$. Moreover, the value at the plateau increases rapidly with $s$. This behaviour corresponds to our comments of section 4.1.

In Figure 5, we have plotted the condition number of the first block $AW_s$ in function of $s$ with $s = 1, 4, 8, 16, 32$ and $m = 96$. We see that it follows an exponential growth, as expected in the symmetric case [2]. Therefore, due to large condition numbers, increasing $s$ past a certain point is a waste of computational time. In a sense, 'The damage has already been done.' Due to this effect, we conclude that a large fixed block size can destroy convergence right after the first block.

## 5.3 Variable $s$-step FibGMRES($m$,$s$) with symmetric matrices

Due to this principle of damage from large $s$ being done quickly when introduced early on in the computation, we advocate a variable block size, where the block size increases gradually. This allows us to properly balance the considerations of the parallel computations that $s$-step affords with the conditioning enhancements brought by a small block size. This approach could be combined with adaptive techniques, where the block size would be chosen according to some indicator. Here we use the Fibonacci sequence described in section 4.2.

### 5.3.1 fv2 matrix with $m = 48$

| $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $s_j$ | 1 | 2 | 3 | 5 | 8 | 13 | 16 |
| $l_j$ | 1 | 3 | 6 | 11 | 19 | 32 | 48 |

Table 2: Variable increasing block size in FibGMRES($m$,$s$) for $m = 48$ and $s = 16$.

| $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $s_j$ | 16 | 13 | 8 | 5 | 3 | 2 | 1 |
| $l_j$ | 16 | 29 | 37 | 42 | 45 | 47 | 48 |

Table 3: Variable decreasing block size in Reverse FibGMRES($m$,$s$) for $m = 48$ and $s = 16$.

We run experiments again with the matrix fv2, $m = 48$, and $s = 16$. We aim to compare the extreme cases $s = 1$ and $s = m = 48$ to the fixed case SGMRES($m$,$s$) with $s = 16$ and to the variable case FibGMRES($m$,$s$) defined by Table 2. Here, we increase gradually the block size following a Fibonacci sequence capped at $s = 16$, and we recall that each new block $B_j$ is of size $s_j$ and that the Krylov size $l_j$ is given by $l_j = l_{j-1} + s_j$. For the sake of comparison, we also gradually decrease the block size from $s = 16$ to $s = 1$ using the Fibonacci sequence in reverse order, as defined in Table 3. We call this algorithm Reverse FibGMRES($m$,$s$).

In Figure 6, convergence of FibGMRES($m$,$s$) is much better than convergence of SGMRES($m$,$s$) and Reverse FibGMRES($m$,$s$). Indeed, we see that Reverse FibGMRES($m$,$s$) defined by Table 3 behaves like SGMRES($m$,$s$), with a hockey-stick pattern. On the other hand, FibGMRES($m$,$s$) convergence curve closely follows the optimal GMRES($m$) curve. As can be seen in Figure 7, the condition number of $AW_{l_j}$ increases gradually with step $j$ for FibGMRES($m$,$s$). On the other hand, at the first step $j = 1$, the condition number $\kappa(AW_{l_1})$ is already large for SGMRES($m$,$s$) and Reverse FibGMRES($m$,$s$). This emphasizes that the faster convergence in FibGMRES($m$,$s$) comes from the increasing sequence, confirming the principle we introduced earlier that the damage can often already be done by the first step of the algorithm. Condition numbers issues must be nipped in the bud early on before they spread.

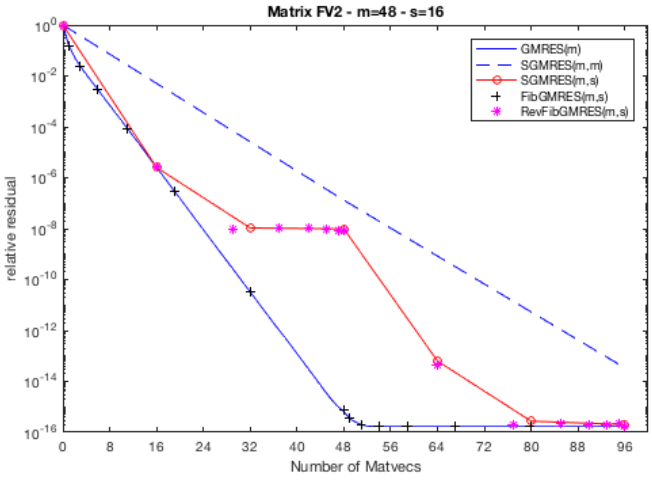


Figure 6: Comparison of fixed and variable $s$-step GMRES. Convergence curves with the matrix fv2, using $m = 48$ and $s = 16$.

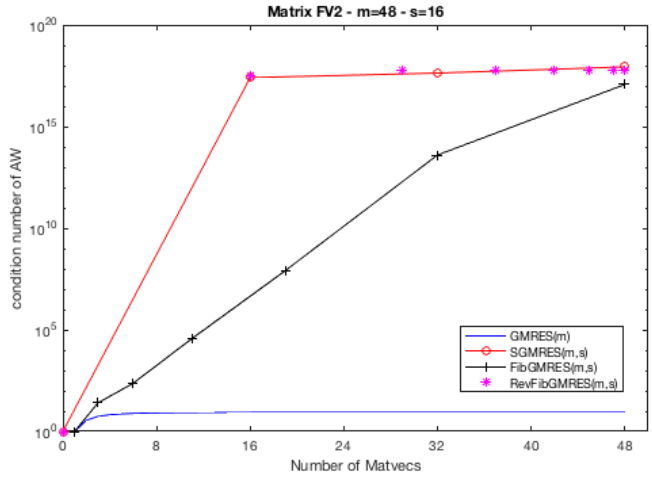


Figure 7: Comparison of fixed and variable $s$-step GMRES. Condition numbers of $AW_{l_j}$ with the matrix fv2, using $m = 48$ and $s = 16$.

### 5.3.2 Poisson matrix with $m = 96$

| $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $s_j$ | 1 | 2 | 3 | 5 | 8 | 13 | 16 | 16 | 16 | 16 |
| $l_j$ | 1 | 3 | 6 | 11 | 19 | 32 | 48 | 64 | 80 | 96 |

Table 4: Variable increasing block size in FibGMRES($m$,$s$) for $m = 96$ and $s = 16$.

We now run experiments with a prototypical test problem, where the matrix arises from a Poisson

| $j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $s_j$ | 1 | 2 | 3 | 5 | 8 | 13 | 14 | 18 | 32 |
| $l_j$ | 1 | 3 | 6 | 11 | 19 | 32 | 46 | 64 | 96 |

Table 5: Variable increasing block size in FibGMRES($m$,$s$) for $m = 96$ and $s = 32$.

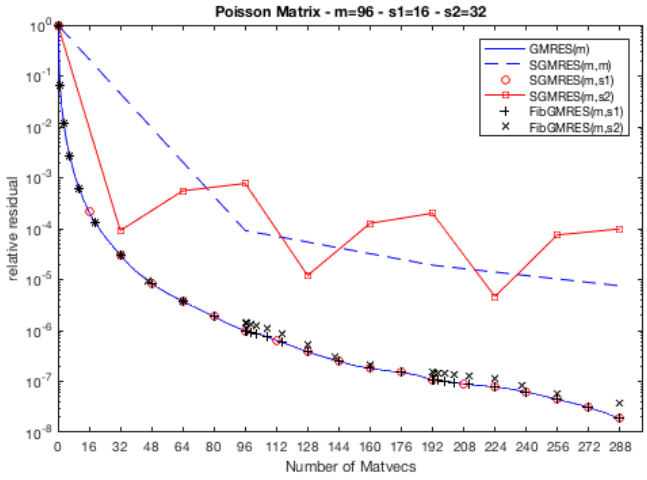


Figure 8: Comparison of fixed and variable $s$-step GMRES. Convergence curves with the Poisson matrix, using $m = 96$ and $s = 16, 32$.

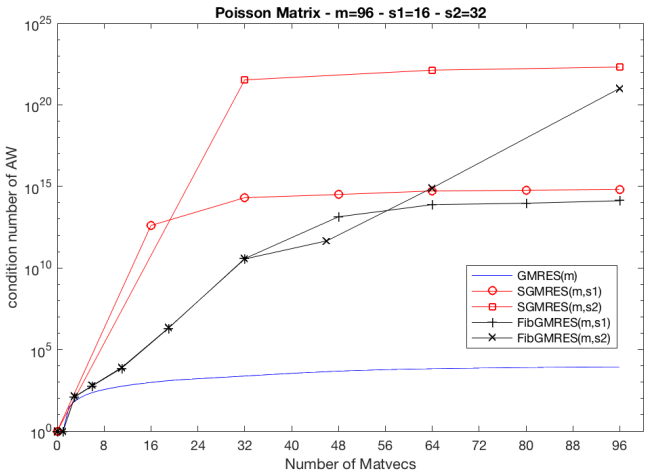


Figure 9: Comparison of fixed and variable $s$-step GMRES. Condition numbers of $AW_{l_j}$ with the Poisson matrix, using $m = 96$ and $s = 16, 32$.

equation in a unit square, discretized by a finited difference scheme, using a 5-point stencil. A regular grid of size $150 \times 150$ results in a sparse matrix of order $n = 22500$. This Poisson matrix is symmetric but not as well-conditioned as the fv2 matrix, so we can expect a slower convergence. We choose a larger

restarting parameter $m = 96$ and two maximal block sizes, namely $s = 16$ and $s = 32$. The sequences of block sizes are defined in Tables 4 and 5. As an aside, we should note that when $s = 32$, the truncated Fibonacci numbers do not neatly add up to $m = 96$. In order to make a fair comparison with fixed s-step GMRES, we have chosen to modify the Fibonacci sequence at step $j = 8$ in order to get $l_j = 96$ at step $j = 9$. In practice, this is not a concern because the user would choose the maximal block size $s$ and the maximal number of steps $J$, in order to get $l_J$ close to a chosen $m$ value.

Convergence results are plotted in Figure 8. For $s = 16$, SGMRES($m$,$s$) and FibGMRES($m$,$s$) converge as quickly as GMRES($m$). However, for $s = 32$, SGMRES($m$,$s$) shows an erratic convergence rate where the residual norms increase and are no longer bounded by those of SGMRES($m$,$m$). On the other hand, FibGMRES($m$,$s$) is still very efficient and converges almost as fast as GMRES($m$). Again, these results can be explained by the condition number $\kappa(AW_{l_j})$. As can be seen in Figure 9, the condition number of the first block ($j = 1$) is quite high with SGMRES($m$,$s$) and then plateaus ($j \geq 2$), with a larger value for $s = 32$ than for $s = 16$. With FibGMRES($m$,$s$), in both cases $s = 16$ and $s = 32$, the condition numbers $\kappa(AW_{l_j})$ increase progressively with $j$. The exponential growth of the condition number is in good agreement with the lower bound of Corollary 1.

### 5.3.3 Large Poisson matrix with $m = 400$

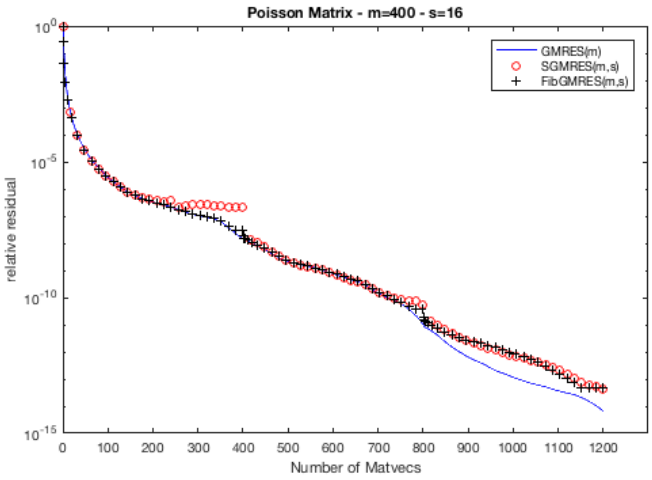


Figure 10: Comparison of fixed and variable $s$-step GMRES. Convergence curves with the large Poisson matrix, using $m = 400$ and $s = 16$.

We now look at a larger Poisson matrix of size $n = 100489$ so that we may extend the Krylov size to $m = 400$, while keeping the block size at $s = 16$. These tests are interesting, because the number of steps is then roughly equal to the quantity $m/s$, as stated in Theorem 4. Thus we can expect similar parallel performance with fixed and variable block size.

In Figure 10, we see that the three algorithms converge roughly at the same rate, but that FibGMRES($m$,$s$) is slighlty faster than SGMRES($m$,$s$) at the end of the first cycle.

## 5.4 Experiments with variable FibGMRES($m$,$s$) and nonsymmetric matrices

The previous experiments were done with symmetric matrices to illustrate various convergence behaviours. Nevertheless, GMRES is designed to solve nonsymmetric systems. Thus we now run experiments with nonsymmetric matrices.

### 5.4.1 cage10 matrix with $m = 48$

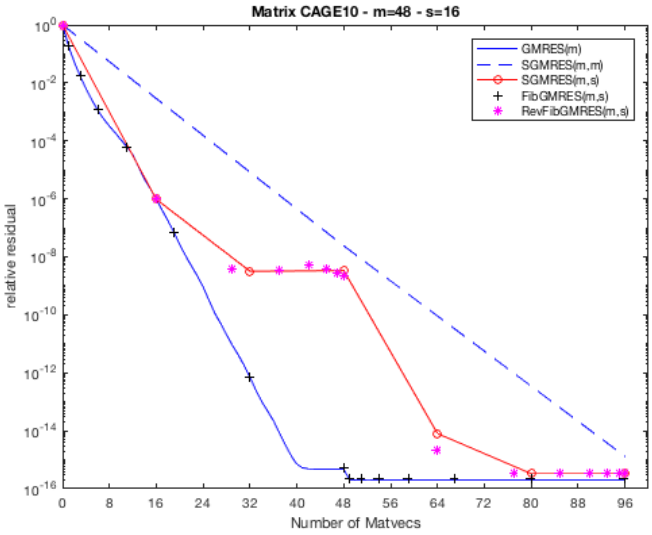


Figure 11: Comparison of fixed and variable $s$-step GMRES. Convergence curves with the matrix cage10, using $m = 48$ and $s = 16$.

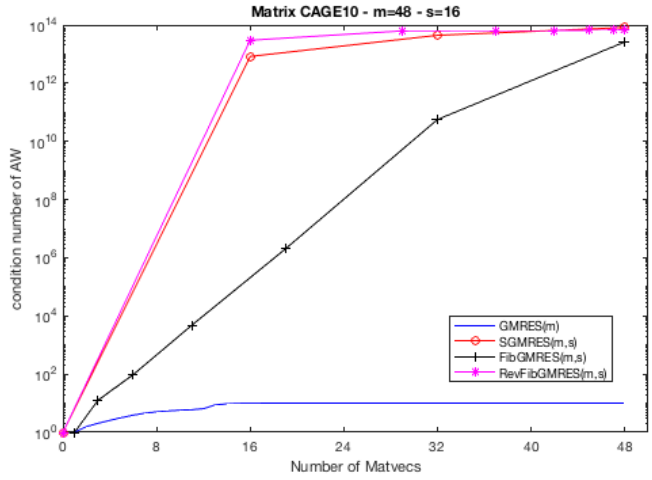


Figure 12: Comparison of fixed and variable $s$-step GMRES. Condition numbers of $AW_{l_j}$ with the matrix cage10, using $m = 48$ and $s = 16$.

The first matrix, called cage10, is from the University of Florida matrix collection [10]. It is of size 11397 with 150645 nonzero elements. Restarted GMRES converges quickly for this matrix, without preconditioning.

We repeat the same tests as for the matrix fv2 and plot the results in Figure 11 and 12. The conclusion is

the same as for fv2: FibGMRES($m$,$s$) converges faster than SGMRES($m$,$s$) and reverse FibGMRES($m$,$s$). Also, condition numbers $\kappa(AW_{l_j})$ corroborate the theoretical results of Corollary 1.

### 5.4.2 PR02R matrix with $m = 96$

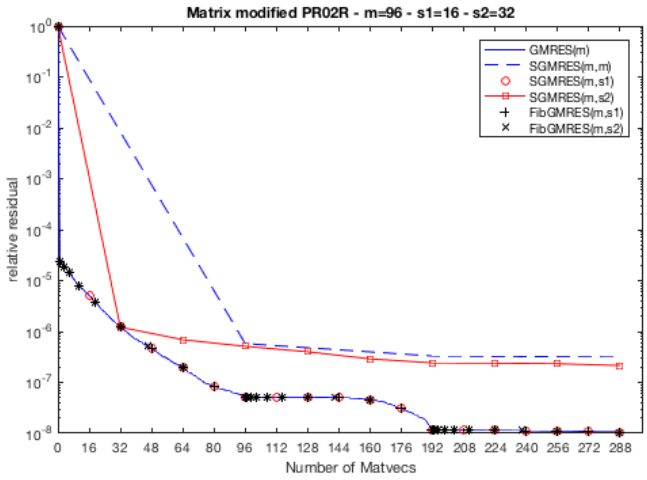


Figure 13: Comparison of fixed and variable $s$-step GMRES. Convergence curves with the modified PR02R matrix, using $m = 96$ and $s = 16, 32$.

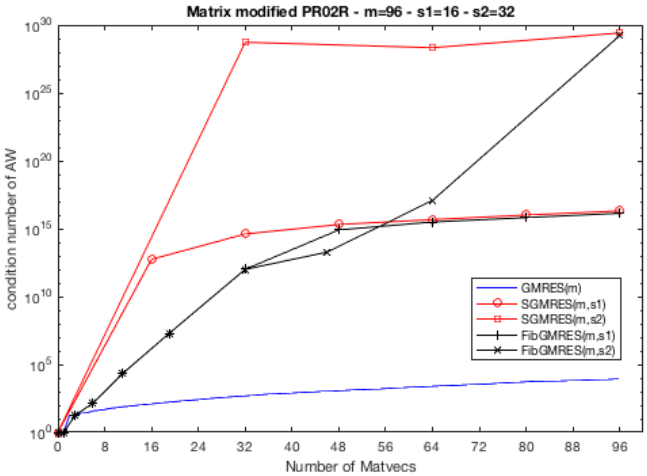


Figure 14: Comparison of fixed and variable $s$-step GMRES. Condition numbers of $AW_{l_j}$ with the modified PR02R matrix, using $m = 96$ and $s = 16, 32$.

The second matrix, PR02R, which is also pulled from the University of Florida matrix collection, represents a turbulence problem from the FLUOREM collection. It is of size 161070 with 8185136 nonzero elements. For this matrix and similar matrices from the FLUOREM collection, iterative methods converge

very slowly or do not converge [37],[29]. Here, in order to get an easier problem with no preconditioning, we have added the matrix $1000 \times I$, where $I$ is the Identity matrix.

We repeat the same experiments as for the Poisson matrix and plot the results in Figures 13 and 14. We observe some stagnation in the second and third cycles of restarted GMRES($m$). Using $s = 16$, both FibGMRES($m$,$s$) and SGMRES($m$,$s$) converge like GMRES($m$), with the same stagnation. But using $s = 32$, FibGMRES($m$,$s$) still converges almost like GMRES($m$), whereas SGMRES($m$,$s$) is almost as slow as SGMRES($m$,$m$). Also, the condition numbers $\kappa(AW_{l_j})$ increase gradually with $j$ for FibGMRES($m$,$s$), whereas the first condition number $\kappa(AW_s)$ is quite large for SGMRES($m$,$s$): as already mentioned, the damage is done.

### 5.4.3 PR02R matrix with $m = 192$

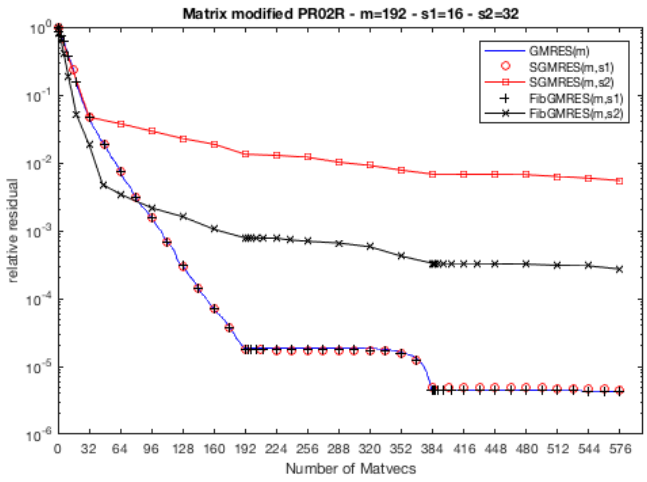


Figure 15: Comparison of fixed and variable $s$-step GMRES. Convergence curves with the modified PR02R matrix, using $m = 192$ and $s = 16, 32$.

Then, we increase the restarting parameter to $m = 192$ with the same block size $s = 16$ or $s = 32$. Here, the ratio $m/s$ is large enough to ensure similar parallel behaviours in the fixed and variable s-step methods. Again, using $s = 16$, the convergence behaviour is similar for restarted, fixed $s$-step and variable $s$-step variants of GMRES. Stagnation still occurs at each restarting cycle except the first one, thus deflation might overcome this issue. Now, using $s = 32$, convergence is damaged for both SGMRES($m$,$s$) and FibGMRES($m$,$s$): the block size becomes too large before restarting. Clearly, an adaptive selection of the block size might avoid this degradation.

# 6 Conclusion

Communication-avoiding Krylov subspace methods are efficient to increase the performance on parallel computers. Among them, $s$-step restarted GMRES, called here SGMRES($m$,$s$), builds an orthonormal basis of a Krylov subspace by consecutive blocks of size $s$. In this paper, we described how to use two bases in order to simplify the algorithm. Then, we proposed to vary the $s$ and defined an original variable $s$-step GMRES algorithm. Our analysis of condition numbers suggests to use an increasing sequence $s_j$ of the block sizes.

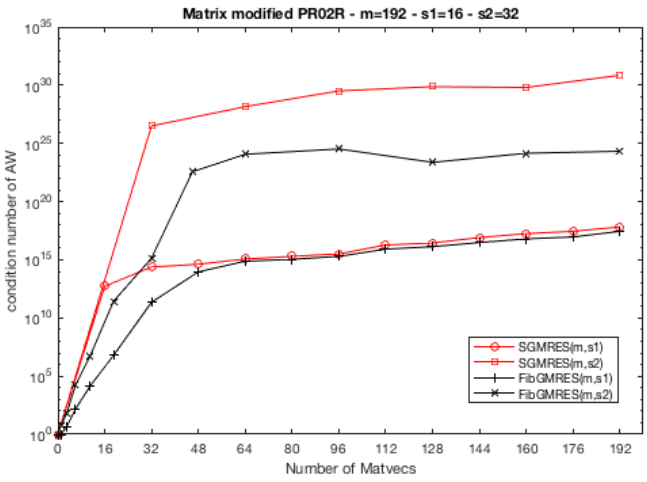


Figure 16: Results with the condition of the modified PR02R matrix, with $m = 192$ and $s = 16, 32$.

In our numerical experiments, we use a Fibonacci sequence, capped at $s$, which avoids communications by limiting the number of steps in the start-up phase. We observe that FibGMRES($m$,$s$) converges often as quickly as GMRES($m$) and faster than SGMRES($m$,$s$).

What we have shown, in essence, is a beneficial tradeoff. An additional cost occurs in the first few steps of the start-up phase of FibGMRES($m$,$s$), because communication overheads are induced by the fan-in fan-out problems associated to small blocks. However, the reduction of condition numbers resulting from FibGMRES($m$,$s$) induces quite often a significant reduction in the number of required iterations.

We could still reduce the number of iterations by introducing a criterion to select adaptively the block size, and by using another basis for each block, for example a Newton basis. In the future, we plan to implement our method on parallel computers, using an efficient matrix-vector product and a communication-avoiding orthogonalization algorithm. Parallel domain decomposition methods combined with deflation will be used for preconditioning the systems. This parallel version will be tested with very large matrices, arising from various computational science problems.

# References


[1] Z. Bai, D. Hu, and L. Reichel, *A Newton Basis GMRES Implementation*, IMA Journal of Numerical Analysis, (1994), pp. 563–581. 14(4).

[2] B. Beckermann, *The condition number of real Vandermonde, Krylov and positive definite Hankel matrices*, Numerische Mathematik, 85 (2000), pp. 553–577.

[3] Bellalij, M., Meurant, G., and Sadok, H., *The Distance of an Eigenvector to a Krylov Subspace and the Convergence of the Arnoldi Method for Eigenvalue Problems*, Linear Algebra and its Applications, (2016), pp. 387–405. 504.

[4] R. H. Bisseling, T. M. Doup, and L. D. J. C. Loyens, *A Parallel Interior Point Algorithm for Linear Programming on a Network of Transputers*, Annals of Operations Research, (1993), pp. 49–86. 43(2).

[5] D. Calvetti, J. Petersen, and L. Reichel, *A Parallel Implementation of the GMRES Method*, in Numerical Linear Algebra, L. Reichel, A. Ruttan, and R. S. Varga, eds., de Gruyter, Berlin, 1993, pp. 31–45.

[6] E. Carson, *Communication-Avoiding Krylov Subspace Methods in Theory and Practice*, PhD thesis, University of California at Berkeley, 2015.

[7] ——, *The adaptive s-step conjugate gradient method*, Tech. Rep. 1701.03989, arXiv, 2017. submitted.

[8] A. T. Chronopoulos, *S-Step Iterative Methods for (non) Symmetric (in) Definite Linear Systems*, SIAM journal on numerical analysis, (1991), pp. 1776–1789. 28(6).

[9] R. D. da Cunha and T. Hopkins, *A Parallel Implementation of the Restarted GMRES Iterative Algorithm for Nonsymmetric Systems of Linear Equations*, Advances in Computational Mathematics, (1994), pp. 261–277. 2(3).

[10] T. A. Davis and Y. Hu, *The University of Florida Sparse Matrix Collection*, ACM Transactions on Mathematical Software (TOMS), (2011). 38(1).

[11] E. De Sturler and H. A. van der Vorst, *Reducing the Effect of Global Communication in GMRES(m) and CG on Parallel Distributed Memory Computers*, Applied Numerical Mathematics, (1995), pp. 441–459. 18(4).

[12] J. Demmel, L. Grigori, M. Hoemmen, and J. Langou, *Communication-optimal Parallel and Sequential QR and LU Factorizations*, SIAM Journal on Scientific Computing, (2012), pp. A206–A239. 34(1).

[13] G. R. Di Brozolo and Y. Robert, *Parallel Conjugate Gradient-like Algorithms for Solving Sparse Nonsymmetric Linear Systems on a Vector Multiprocessor*, Parallel Computing, (1989), pp. 223–239. 11(2).

[14] J. Erhel, *A Parallel GMRES Version for General Sparse Matrices*, Electronic Transaction on Numerical Analysis, (1995), pp. 160–176.

[15] L. Giraud, J. Langou, M. Rozloznik, and J. van den Eshof, *Rounding Error Analysis of the Classical Gram-Schmidt Orthogonalization Process*, Numerische Mathematik, (2005), pp. 87–100. 101(1).

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations (4th edition)*, John Hopkins University Press, 2013. (Vol. 4). JHUP.

[17] L. Grigori and S. Moufawad, *Communication avoiding ILU0 preconditioner*, SIAM Journal on Scientific Computing, 37 (2015), pp. C217–C246.

[18] N. J. Higham, *Accuracy and Stability of Numerical Algorithms (second edition)*, SIAM, 2002.

[19] M. F. Hoemmen, *Communication-avoiding Krylov subspace methods*, PhD thesis, EECS Department, University of California, Berkeley, Apr 2010.

[20] A. S. Householder, *The Theory of Matrices in Numerical Analysis*, Dover Publications., 1964.

[21] W. Jalby and B. Philippe, *Stability Analysis and Improvement of the Block Gram-Schmidt Algorithm*, SIAM journal on scientific and statistical computing, (1991), pp. 1058–1073. 12(5).

[22] W. D. Joubert and G. F. Carey, *Parallelizable restarted iterative methods for non- symmetric linear systems, Part I: Theory*, International Journal of Computer Mathematics, 44 (1992), pp. 243–267.

[23] ——, *Parallelizable restarted iterative methods for non- symmetric linear systems, Part II: Parallel implementation*, International Journal of Computer Mathematics, 44 (1992), pp. 269–290.

[24] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, 1995.

[25] S. Kim and A. T. Chronopoulos, *An Efficient Nonsymmetric Lanczos Method on Parallel Vector Computers*, Journal of Computational and Applied Mathematics, (1992), pp. 357–374. 42(3).

[26] M. Mohiyuddin, M. Hoemmen, J. Demmel, and K. Yelick, *Minimizing Communication in sparse matrix solvers*, Proceedings of the Conference on High Performance Computing Networking. Storage and Analysis, (2009), p. 36.

[27] N. Nassif, J. Erhel, and B. Philippe, *Introduction to Computational Linear Algebra*, CRC Press, 2015.

[28] D. Nuentsa Wakam and J. Erhel, *Parallelism and Robustness in GMRES with the Newton Basis and the Deflated Restarting*, Electronic Transactions on Numerical Analysis, 40 (2013), pp. 381–406.

[29] D. Nuentsa Wakam and F. Pacull, *Memory efficient hybrid algebraic solvers for linear systems arising from compressible flows*, Computers and Fluids, 80 (2013), pp. 158–167.

[30] B. Philippe and L. Reichel, *On the generation of Krylov subspace bases*, Applied Numerical Mathematics (APNUM), 62 (2012), pp. 1171–1186.

[31] B. Phillipe and R. Sidje, *Parallel Algorithms for the Arnoldi Procedure.*, Iterative Methods in Linear Algebra, II, IMACS Ser. Comput. Appl. Math, (1994), pp. 156–165. 3.

[32] K. Rosen, *Elementary Number Theory*, Pearson Education., 2011.

[33] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM., 2003.

[34] Y. Saad and M. H. Schultz, *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*, SIAM Journal on scientific and statistical computing, (1986), pp. 856–869. 7(3).

[35] J. N. Shadid and R. S. Tuminaro, *Sparse Iterative Algorithm Software for Large Scale MIMD Machines: An initial discussion and implementation*, Concurrency: practice and experience, (1992), pp. 481–497. 4(6).

[36] R. B. Sidje, *Alternatives for Parallel Krylov Basis Computation*, Numerical Linear Algebra with Applications, 4 (1997), pp. 305–331.

[37] D. N. Wakam, J. Erhel, E. Canot, and G. Atenekeng-Kahou, *A comparative study of some distributed linear solvers on systems arising from fluid dynamics simulations*, in Parallel Computing: from Multicores and GPU's to Petascale (proceedings of PARCO'09), B. Chapman, F. Desprez, G. Joubert, A. Lichnewsky, F. Peters, and T. Priol, eds., vol. 19 of Advances in parallel computing, IOS Press, 2010, pp. 51–58.

[38] H. F. Walker, *Implementation of the GMRES Method Using Householder Transformations*, SIAM Journal on Scientific and Statistical Computing, (1988), pp. 152–163. 9(1).

[39] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford Science Publications., 1965.