

The Influence of Reordering Algorithms on the Convergence of a Preconditioned Restarted GMRES Method

S. L. Gonzaga de Oliveira[1(✉)], C. Carvalho[1], and C. Osthoff[2]

[1] Universidade Federal de Lavras, Lavras, MG, Brazil
sanderson@ufla.br
[2] Laboratório Nacional de Computação Científica, Petropólis, RJ, Brazil
osthoff@lncc.br

Abstract. This paper concentrates on applying reordering algorithms as a preprocessing step of a restarted Generalized Minimal Residual (GMRES for short) solver preconditioned by three ILU-type preconditioners. This paper investigates the effect of 13 orderings on the convergence of the preconditioned GMRES solver restarted every 50 steps when applied to nine real large-scale nonsymmetric and not positive definite matrices. Specifically, this paper shows the most promising combination of preconditioners and reordering for each linear system used.

Keywords: Bandwidth reduction · Profile reduction · Reordering algorithms · Graph labeling · GMRES method

1 Introduction

This paper focus on the solution of linear systems in the form $Ax = b$, where $A = [a_{ij}]$ is an $n \times n$ large-scale sparse matrix, x is the unknown n-vector solution, and b is a known n–vector. The solution of linear systems is an important step in various scientific and engineering simulations. The solution of linear systems comprised of large-scale matrices requires long computing times. Specifically, this paper show experiments using real nonsymmetric (structurally as well as numerically) matrices that are not positive definite. As examples, these matrices emerge from the circuit simulation, electromagnetics, and optimization problems. These problems can cause serious complications for preconditioned iterative methods [1,2].

Iterative methods are applied to solve large-scale problems in scientific computing. Since the sizes of the problems in this study are larger than 400,000 unknowns, we applied an iterative method to solve the linear systems. The GMRES method of Saad and Schultz [3] is a leading Krylov subspace method for solving non-Hermitian linear systems $Ax = b$, where $A \in \mathbb{C}^{n\times n}$ is a non-singular matrix and $b \in \mathbb{C}^n$ (see [1] and references therein). The method uses the Arnoldi algorithm. Hence, its execution costs grow with each iteration. Thus,

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12249, pp. 19–32, 2020.
https://doi.org/10.1007/978-3-030-58799-4_2

researchers typically restart the GMRES method every m iterations [3], using the last computed residual as the next initial one. This restarted approach may stagnate [3]. Even if stagnation does not occur, convergence can be slow (see [1]). Hence, preconditioning and data preprocessing are fundamental steps for the use of the restarted GMRES solver.

It is common sense that if the matrix has nonzero coefficients close to the main diagonal, the number of floating-point operations in the ILU factorization can be smaller. Consequently, reordering the unknowns and equations influence the rate of convergence of preconditioned Krylov subspace methods [4,5]. We studied the potential of the preconditioned restarted GMRES method with symmetric permutations of the matrix A for solving linear systems. We used the structure of $A + A^T$ in the experiments. Specifically, we investigated the effects of 13 symmetric permutations to further improve incomplete (approximate) factorization preconditioning for sequential calculations with the restarted GMRES(50) solver. Incomplete factorization preconditioners also depend on the ordering of unknowns and equations.

There are important reasons for finding an appropriate order for the row and columns of the coefficient matrix A. Modern hierarchical memory architecture and paging policies favor programs that consider the locality of reference into account. Heuristics for bandwidth and profile reductions are usually applied to yield an arrangement of graph (corresponding to the matrix A) vertices with the spatial locality. Therefore, these heuristics are typically employed to achieve low running times for solving large-scale sparse linear systems by iterative methods [6].

The present study employs the restarted GMRES(50) solver preconditioned by three preconditioners: ILUT [7], ILUC [8], and Algebraic Recursive Multilevel Solver (ARMS) [9]. Specifically, we employed in the experiments these preconditioners and the Flexible GMRES solver available in ITSOL v.2.0, a package of iterative solvers [10]. Particularly, this is a state-of-the-art implementation of the restarted GMRES method and its preconditioners.

This paper shows a set of numerical experiments on the influence of 13 low-cost reordering strategies on the convergence of the GMRES solver restarted every 50 steps when using three ILU-type preconditioners for a set of nine large-scale sparse matrices. The matrices are not positive definite, but real nonsymmetric matrices.

We selected low-cost heuristics for bandwidth and profile reductions because an adequate vertex labeling of a graph associated with a matrix contained in a linear system may reduce the computational times of an iterative linear solver, such as the GMRES method [4,5]. However, the bandwidth and profile reductions yielded by the reordering algorithms are not directly proportional to the computational time reduction of solving linear systems by an iterative solver. Furthermore, at least when the simulation solves only a single linear system, the total running time, including the time of the reordering procedure, is to be minimized. Thus, an application should employ a low-cost reordering algorithm [6,11]. The reordering strategies evaluated in this study are graph-theory

algorithms that perform only symmetric permutations, preserve the set of diagonal entries, and neither affect the coefficients nor the eigenvalues of the nonsymmetric matrix A.

The remainder of this manuscript is structured as follows. In Sect. 2, important works in this field are reviewed. In Sect. 3, we describe how the experiments were conducted in this study. In Sect 4, we comment on the parameters evaluated in the preconditioned restarted GMRES solver. In Sect. 5, the results and analyses are presented. Finally, in Sect. 6, the conclusions are addressed.

2 Related Work

A previous publication reviewed the most important works in this context [1]. We provide only a brief review of the most important contributions in the field.

Benzi *et al.* [4] performed simulations with reordering nonsymmetric matrices for solving linear systems by the GMRES method. Camata *et al.* [5] investigated the use of ordering strategies in tandem with the ILU algorithm used as a preconditioner to the restarted GMRES solver. The authors compared the results yielded by the natural, quotient minimum degree, and RCM-GL [12] orderings. In general, Benzi *et al.* [4] and Camata *et al.* [5] recommended employing the RCM-GL [12] as a preprocessing step for the preconditioned restarted GMRES solver. In particular, the RCM-GL method yielded better results than other methods when applied to reduce the execution times of a linear system solver [13,14].

A recent publication [1] evaluated the effect of nine heuristics for bandwidth reduction on the convergence of the GMRES(50) solver preconditioned by the ILU, ILUT, VBILU, and VBILUT algorithms when applied to 11 linear systems. In the present study, we employed the same solver preconditioned by two other algorithms (ILUC and ARMS, as well as the ILUT preconditioner) applied to nine different linear systems. Additionally, the present study employs four different reordering strategies.

Previous publications (see [6] and references therein) recognized seven promising low-cost heuristics for bandwidth and profile reductions: KP-band [15], reverse breadth-first search (BFS) with starting vertex given by the George-Liu (GL) algorithm (see [6]), RCM-GL [12], MPG [16], Sloan [17], NSloan [18], and Sloan-MGPS [19] heuristics. Similar to the previous work [1], this paper also incorporate in the experiments the BFS, reverse BFS [4], BFS-GL, Cuthill-McKee (CM) [20], Reverse Cuthill-McKee (RCM) [21], and CM-GL procedures.

3 Description of the Tests

We wrote the codes of the heuristics for bandwidth and profile reductions in the C++ programming language using the g++ version 5.4.0 compiler, with the optimization flag -O3. To evaluate the performance of the preconditioned restarted GMRES solver [3] computed after executing reordering algorithms, we selected real nonsymmetric linear systems ranging from 400,000 to 1,000,000 unknowns contained in the SuiteSparse matrix collection [22].

In the experiments, the GMRES(50) solver stopped either when the norm of the residual vector was less than 1e-8 or when a maximum number of iterations (n) was reached. Since the preconditioned GMRES method [10] with 50 as the restart parameter is a fast iterative solver, we aborted an execution that computed more than 20 min supposing that either the solver would not converge or the solver would converge after a much longer time. When using the more difficult linear systems, however, a timeout of 3,600 s was established. The solver executed if the condest associated with the matrix was smaller than 1e+15.

The workstation used in the execution of the simulations featured an Intel® Core™ i7-4770 (16 GB of main memory DDR3 1.333 GHz) (Intel; Santa Clara, CA, United States). The machine used the Ubuntu 16.04.3 LTS 64-bit operating system with Linux kernel-version 4.13.0-39-generic.

4 Parameters Evaluated in the Preconditioned Restarted GMRES Solver

In this section, we show how we chose the initial parameters in the restarted GMRES solver preconditioned by the ILUT, ILUC, and ARMS preconditioners. A previous publication [1] showed that the best results were achieved when using the preconditioned restarted GMRES solver with 50 vectors in the Krylov basis, denoted GMRES(50) solver, among the three other parameters evaluated (30, 100, and 250). With this value for the GMRES solver, we evaluated the most suitable parameters on average for the three preconditioners when applied to the same 35 linear systems (with sizes ranging from 12,111 to 150,102).

Table 1 shows the parameters used for the three preconditioners evaluated in this study. We considered that the best combinations of parameters are those that led to the convergence of the GMRES solver at the lowest cost.

Table 1. Preliminary parameters studied in the preconditioned restarted GMRES solver [3,10].

Method	Parameter	Values
GMRES	m	30, 50, 100, 250
ILUT,	τ	1e-1, 1e-3, 1e-6, 1e-9
ILUC	p	15, 50, 100, max. degree
ARMS	Number of levels	3, 5, 10
	Tol_{dd}	0.1, 0.4, 0.7
	Block size	30, 100, $\mathtt{m}$

The choice of parameters may not be ideal for every matrix used due to the variety of characteristics present in the linear systems used in this study (i.e., the large number of unknowns, large number of non-null coefficients, etc.). Furthermore, the linear systems arise from different application areas. This generic

choice of parameters means that a generic application of the preconditioned restarted GMRES solver will be unsuccessful for some instances contained in the dataset used. Nevertheless, these values made effective the set of algorithms for eight out of nine linear systems used in this study, recalling that the objective was to investigate the performance of reordering algorithms employed as a preprocessing step for the preconditioned restarted GMRES solver. Thus, we decided to set a "generic" preconditioned restarted GMRES solver to study the performance of the reordering algorithms.

For each parameter, we calculated the average times of the preconditioned restarted GMRES solver. Then, we calculated the metric $\rho = \sum_{i=1}^{N} \frac{t_H(i) - t_{min}(i)}{t_{min}(i)}$ [11] with these average times, where $t_H(i)$ is the execution time yielded when using an algorithm H applied to the instance i, $t_{min}(i)$ is the lowest computing time achieved in the instance i, and N is the number of instances.

Usually, searching for satisfactory values, researchers empirically chose the parameters τ and p for a few sample matrices from a specific application. The optimal values are heavily problem-dependent. Similar to the previous publication [1], exploratory investigations showed that the preconditioners yielded better results when using the parameter τ established as 1e-3 both when using the results yielded by applying only the GMRES(50) solver and the solver with the parameters used in the previous publication [1]. The experiments also showed that the preconditioners yielded better results when using the parameter p established as 50. Thus, we employed the ILUT and ILUC algorithms with the parameters (1e-3, 50), (1e-6, 75), (1e-9, 100), and (1e-15,250) when preconditioning the GMRES(50) solver. We used the parameters (1e-15,250) when the preconditioned iterative linear solver did not converge with the other parameters.

The ARMS preconditioner available in the ITSOL package [10] has three parameters: number of levels, a threshold value used for the relative diagonal dominance (tol_{dd}) criterion (ranging from 0 to 1), and block size [10]. The ρ values in Table 2 shows that the best results were achieved when defining the parameters in the ARMS preconditioner related to number of levels, tol_{dd}, and block size as ARMS(5,0.4,30) and ARMS$(3, 0.1, \mathfrak{m})$, where $\mathfrak{m} = \left\lfloor \max_{v \in V}[degree(v)]/2 \right\rfloor$, in two cases: when m is equal to i) 30, 50, 100, and 250, and ii) 50, respectively. Considering the size of the linear systems used in the experiments, we defined the number of levels as 5 to apply ARMS(5,tol_{dd},30) with tol_{dd} set as 0.1, 0.4, and 0.7, and ARMS$(5, 0.7, \mathfrak{m})$ preconditioners.

Table 3 shows the definitive parameters that we used for the preconditioners. We used the parameters described in the following round if the preconditioned GMRES(50) solver did not converge with the parameters set in the previous experiment.

5 Results and Analysis

Tables 4 and 5 show the instance's name, preconditioner used, the average running time of the preconditioned GMRES(50) solver without the use of a reordering algorithm (see column Original), and the running times of this solver in

Table 2. Metric ρ calculated for several combinations of parameters in the ARMS algorithm when preconditioning the GMRES solver [3,10].

GMRES(m)	No. levels	3	5	10
30,50,100,250	ρ	4	**2**	31
50	ρ	**4**	8	30
GMRES	Tol_{dd}	0.1	0.4	0.7
30,50,100,250	ρ	10.8	**10.6**	48.9
50	ρ	**4.2**	21.5	47.9
GMRES	Block size	30	100	m
30,50,100,250	ρ	**5.1**	5.2	8.4
50	ρ	26.0	4.2	**3.4**

Table 3. Parameters used in the experiments with the preconditioned GMRES(50). The program applied values of the following trial only to linear systems that the solver did not achieve convergence in the previous experiment.

Preconditioner	Parameter	Round			
		1st	2nd	3rd	4th
ILUT,	τ	1e-3	1e-6	1e-9	1e-15
ILUC	p	50	75	100	250
ARMS	Tol_{dd}	0.1	0.4	0.7	0.7
	Nr. of levels	5			
	Block size	30			m

tandem with several reordering algorithms. The symbol * in these tables indicates that the solver did not converge because of some inconsistency during preconditioning, such as a diagonal or pivot null or exceedingly small pivots, which is a possibility for matrices that do not have diagonal dominance and for highly unstructured problems. This kind of failure occurs when the preconditioner drops many large fill-ins from the incomplete factors. Ill-conditioning of the triangular factors also results in the instability of the recurrences involved in the forward and backward solves when executing the preconditioning [4]. Additionally, "cond" indicates that we aborted the execution because of the condest is greater than 1e+15. The symbol † in the same tables means that we stopped the run because it computed for 20 min. A superscript indicates the trial that the solver converged. It also means that the solver did not converge in the previous experiment(s). The symbol ‡ denotes that we performed the fourth trial with the solver applied to the linear system, and the simulation terminated for exceeding 3,600 s. Numbers in light gray color indicate the run that required the least amount of work when using the preconditioner. This case is not always the same as the run that demanded the lowest number of iterations. The reason is that, in general, different orderings result in a preconditioner with a different

Table 4. Execution times (s) of the preconditioned GMRES(50) solver applied to five linear systems containing nonsymmetric matrices.

	Preconditioner		Original	RBFS-GL	RCM-GL	RBFS	RCM	BFS-GL	CM-GL	BFS	CM	KP	NSloan	Sloan	Sloan-MGPS	MPG
rajat21	β		411571	410770	411032	410023	406325	410770	411032	410023	406325	411571	286495	286495	286495	-
	Profile($\cdot 10^6$)		20769	1285	2351	3259	4275	1285	2351	3259	4275	3703	3869	3869	3869	
	ILUT		*	*	*	*	*	*	*	*	*	*	*	*	*	*
	ILUC		‡	‡	cond	‡	cond	cond	‡	cond	‡	cond	cond	cond	cond	‡
	ARMS	t(s)	$9.4^{/4}$	$14.6^{/4}$	$13.9^{/4}$	$25.2^{/4}$	$12.9^{/4}$	$12.0^{/4}$	$20.9^{/4}$	$11.0^{/4}$	$27.1^{/4}$	*	*	*	*	‡
largebasis	β		400019	17	17	19	19	*	17	19	19	21	39	36	39	41
	Profile($\cdot 10^6$)		72006	10	10	10	10	*	10	10	10	11	13	12	13	12
	ILUT	t(s)/iter	*	4.0/12	*	3.9/24	*	*	8.5/24	*	7.4/24	8.8/22	*	*	*	*
	ILUC		cond	†	†	†	†	†	†	†	†	cond	†	†	†	†
	ARMS	t(s)	$136.9^{/3}$	$371.7^{/3}$	$1115.1^{/3}$	$386.0^{/3}$	$65.0^{/3}$	$62.7^{/3}$	746.6	$61.5^{/3}$	$389.7^{/3}$	699.7	$72.8^{/3}$	$76.1^{/3}$	$12.1^{/3}$	$107.9^{/3}$
		iter.	13	14	7	7	12	12	71	12	7	49	5	11	5	12
cage13	β		206333	43926	67522	66483	68262	43267	43699	66483	68262	45337	244025	244255	158310	261828
	Profile($\cdot 10^6$)		17473	1943	2273	2115	2277	1936	2146	2115	2277	2999	1680	655	2986	582
	ILUT	t(s)	8.3	16.0	21.6	13.5	18.3	17.0	20.7	13.6	18.4	23.2	24.5	24.7	24.6	225.6
	(1e-3,50)	iter.	4	4	4	4	4	4	4	4	4	4	4	4	4	4
	ILUC	t(s)	5.4	13.3	17.3	9.3	14.2	12.4	17.0	9.4	14.0	18.0	20.2	19.2	19.3	219.9
	(1e-3,50)	iter.	15	15	15	15	15	15	15	15	15	15	15	15	15	15
	ARMS		729.7	†	†	†	†	†	†	†	†	†	†	†	†	†
rajat29	β		643985	638910	638900	628692	621471	638910	638900	628692	621471	638901	-			
	Profile($\cdot 10^6$)		154707	7350	4251	4234	1061	7350	4251	4234	1061	816				
	ILUT		*	*	*	*	*	*	*	‡	*	*	‡	‡	‡	‡
	ILUC		‡	‡	cond	‡	cond	cond	‡	‡	‡	cond	‡	‡	‡	‡
	ARMS	t(s)	$133.5^{/4}$	$56.3^{/4}$	$30.8^{/4}$	$54.3^{/4}$	$29.1^{/4}$	$246.8^{/4}$	$66.9^{/4}$	$157.0^{/4}$	$65.2^{/4}$	*	‡	‡	‡	‡
rajat30	β		643985	641890	641882	618146	608973	641890	641882	618146	608973	641883	-			
	Profile($\cdot 10^6$)		154740	3029	2740	4076	2281	3029	2740	4076	2281	3437				
	ILUT		*	*	*	*	*	*	*	†	*	*	†	†	†	†
	ILUC		†	†	cond	†	cond	cond	†	cond	†	cond	†	†	†	†
	ARMS	t(s)	228.1	85.3	93.4	81.2	91.3	350.8	90.1	319.7	87.9	$359.9^{/3}$	‡	‡	‡	‡
		iter	56	57	57	59	57	59	58	59	57	56				

Table 5. Execution times (s) of the preconditioned GMRES(50) solver applied to four linear systems containing nonsymmetric matrices.

	Preconditioner		Original	RBFS-GL	RCM-GL	RBFS	RCM	BFS-GL	CM-GL	BFS	CM	KP	NSloan	Sloan	Sloan-MGPS	MPG
pre2	β		117686	122060	106715	111213	106715	194193	136572	180310	202917	167760	122676	122676	122676	102002
	Profile($\cdot 10^6$)		7034	3606	1282	1602	1282	1843	1313	1602	2849	2253	3990	3990	3990	4228
	ILUT		*	*	*	*	*	*	*	*	*	*	*	*	*	*
	ILUC		cond	cond	cond	cond	cond	cond	cond	cond	cond	cond	cond	cond	cond	cond
	ARMS		‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡	‡
ASIC_680ks	β		681277	124668	152927	108366	200311	124668	152927	108366	112761	149106	640088	640088	640088	629324
	Profile($\cdot 10^6$)		3728	3425	1895	1895	2415	3728	3425	1895	2415	1400	143	143	143	842
	ILUT	t(s)	0.3	2.4	3.4	1.4	2.8	3.0	3.4	2.4	2.1	3.9	7.9	8.0	8.0	53.1
	(1e-3,50)	iter.	2	4	2	4	12	13	2	20	4	7	23	18	18	12
	ILUC		†	†	†	†	†	†	†	†	†	†	†	†	†	†
	ARMS	t(s)	14.7	3.6	11.7	2.5	$3.3^{/2}$	3.7	4.3	2.8	3.3	4.8	11.5	12.0	12.0	$54.8^{/2}$
		iter.	121	6	11	6	7	5	6	7	6	9	12	12	12	9
ASIC_680k	β		682779	682384	682389	682383	682385	682384	682389	682383	682385	682389				
	Profile($\cdot 10^6$)		147368	3425	4154	550	4139	552	4154	550	4139	808				
	ILUT	t(s)/iter.	*	*	*	*	6.9/4	*	*	4.2/25	*	*	†	†	†	†
	ILUC		†	*	†	*	†	†	†	†	†	†	†	†	†	†
	ARMS	t(s)	141.6	182.2	187.1	180.8	185.5	132.7	158.3	131.2	157.0	187.1	†	†	†	†
		iter.	19	5	6	5	6	5	5	5	6	5				
tmt_unsym	β		2161	948	946	1081	1081	948	946	1081	1081	948	26926	1659	1654	1528
	Profile($\cdot 10^6$)		842	1278	1277	1535	1535	1278	1277	1535	1535	1278	1184	1277	1277	1192
	ILUT	t(s)	$90.8^{/2}$	$80.0^{/2}$	$87.2^{/2}$	$77.5^{/2}$	$85.2^{/2}$	$82.7^{/2}$	$84.9^{/2}$	$80.9^{/2}$	$82.4^{/2}$	$90.3^{/2}$	$90.3^{/2}$	$84.4^{/2}$	$84.3^{/2}$	$82.8^{/2}$
	(1e-6,75)	iter.	50	39	42	39	42	42	40	42	39	42	48	39	39	39
	ILUC, ARMS		†	†	†	†	†	†	†	†	†	†	†	†	†	†

number of non-null coefficients [4]. Results in dark gray color are the best results achieved in the linear system. This case is also not necessarily the same as the run that required the lowest number of iterations.

This GMRES(50) solver only converged when applied to the *rajat21*, *rajat29*, and *rajat30* instances using the ARMS preconditioner. The GMRES(50) solver preconditioned with the same preconditioner yielded the best results when applied to the original ordering of the instance *rajat21*. The GMRES(50) solver preconditioned with the ARMS(5,0.7,$\mathfrak{m}$) algorithm in tandem with the RCM ordering delivered the best results when applied to the instance *rajat29*. The GMRES(50) solver preconditioned with the ARMS(5,0.1,30) algorithm in conjunction with the BFS ordering yielded the best results when applied to the instance *rajat30*.

Table 4 shows that the RBFS ordering in conjunction with the ILUT(1e-3,50) preconditioner yielded the best results (3.9 s) when using the GMRES(50) solver applied to the instance *largebasis*. The simulations with the GMRES(50) solver preconditioned with the ARMS(5,0.7,30) algorithm converged with the 13 orderings evaluated in this study.

Table 4 also shows that the three preconditioners evaluated in tandem with the GMRES(50) solver yielded better results when applied to the original ordering of the instance *cage13* than employing the 13 reordering algorithms analyzed in the present study. The ILUC(1e-3,50) preconditioner delivered the best results when applied to this linear system.

Despite the number of preconditioners used with different sets of parameters (listed in Table 3) and the use of 13 orderings, the solver did not converge when applied to the instance *pre2*. One could examine better parameters for the preconditioned restarted GMRES solver to achieve convergence with this linear system. However, the principal objective was to study symmetrical permutations used as a preprocessing step of this preconditioned iterative solver.

The GMRES(50) solver preconditioned with the ILUT(1e-3,50) algorithm yielded the best results to the instance *ASIC_680ks* applied to the original ordering of the instance. The GMRES(50) solver preconditioned with the ILUT(1e-3,50) algorithm in tandem with the BFS ordering yielded the best results when applied to the instance *ASIC_680k*.

Table 5 shows the results of the algorithms applied to the instance *tmt_unsym*. The GMRES(50) solver only converged to this instance when preconditioned with the ILUT(1e-6,75) preconditioner. The simulation with the RBFS ordering yielded the shortest running time when applied alongside the ILUT(1e-6,75) preconditioner.

Table 6 provides characteristics (name, size, number of non-null coefficients ($|E|$), application area, the best preconditioner employed, the best ordering applied to the linear system, and the best time computed) of the linear systems composed of real nonsymmetric matrices used in this computational experiment. The RBFS ordering [4] benefited the convergence of the GMRES(50) solver when applied in conjunction with the ILUT [(1e-3,50) and (1e-6,75)] preconditioner in two cases (instances *largebasis* and *tmt_unsym*). The GMRES(50)

solver only converged to the instance *tmt_unsym* when applied in tandem with the ILUT preconditioner. Additionally, the RBFS [RCM] ordering favored the convergence of the GMRES(50) solver in conjunction with the ARMS(5,0.1,30) [ARMS(5,0.7,m)] preconditioner when applied to the instance rajat30 [rajat29]. In particular, the use of a reordering algorithm favored the preconditioned GMRES(50) solver to converge much faster than using the original ordering when applied to the instances *largebasis*, *rajat29*, *rajat30*, and *ASIC_680k*. Table 6 highlights these instances in gray color. The BFS procedure had a positive effect when applied to the instance *ASIC_680k* regarding the convergence of the GMRES(50) solver with the ILUT(1e-3,50) preconditioner.

Table 6. Best orderings and preconditioners used in conjunction with the GMRES(50) solver.

Instance	Size	$\|E\|$	Preconditioner	Ordering	t(s)
rajat21	411676	1876011	ARMS(5,0.7,m)	Original	9.0
largebasis	440020	5240084	ILUT(1e-3,50)	RBFS	3.9
cage13	445315	7479343	ILUC(1e-3,50)	Original	5.3
rajat29	643994	3760246	ARMS(5,0.7,m)	RCM	29.1
rajat30		6175244	ARMS(5,0.1,30)	RBFS	81.2
pre2	659033	5834044	—	—	—
ASIC_680ks	682712	1693767	ILUT(1e-3,50)	Original	0.3
ASIC_680k	682862	2638997		BFS	4.2
tmt_unsym	917825	4584801	ILUT(1e-6,75)	RBFS	77.5

The GMRES(50) solver, when preconditioned by at least one of the three preconditioners employed in this study and applied to the original ordering of the linear system, solved all the eight test problems computed. There was no need to circumvent convergence through reordering in any of these eight problems. Table 6 shows that the GMRES(50) solver yielded the best results when employing a specific preconditioner with the original ordering itself in three out of these nine linear systems. Additionally, the experiments showed that the use of the preconditioned GMRES(50) solver applied to the original sequence of unknowns and equations of the linear system yielded similar times to the best results when a reordering algorithm was employed in simulations using the instance *tmt_sym*.

The experiments showed that the number of iterations is not the best metric to analyze the results. For example, when applied to the instance *largebasis*, the solver in tandem with the ILUT preconditioner took 4.0 and 3.9 s with the RBFS-GL and RBFS orderings in 12 and 24 iterations, respectively. The solver in tandem with the ARMS preconditioner, when applied to the instance *rajat30*, took respectively 81 and 360 s with the RBFS and KP orderings in 59 and 56 iterations. When applied to the instance *ASIC_680k*, the solver in conjunction

with the ILUT preconditioner took 4 and 7 s with the BFS and RCM orderings in 25 and 4 iterations, respectively.

It is common sense that an important metric to be evaluated is the bandwidth of the matrices before and after the reordering. In this sense, reordering cannot be useful for preprocessing matrices that have small bandwidth. For example, the reordering methods did not reduce enough the bandwidth of the instance *pre2*. The solver did not converge when applied to this linear system. On the other hand, although the reordering algorithms did not reduce enough the bandwidth of the instances *rajat29*, *rajat30*, *ASIC_680k*, the reordering algorithms favored the computing times of the restarted GMRES solver. Nevertheless, in three cases (*cage13*, *rajat21*, *ASIC_680ks*), although the heuristics reduced the bandwidth of the matrices, the best results were achieved with the original ordering.

The GMRES(50) solver preconditioned by the ARMS preconditioner converged to seven out of nine linear systems used in this study. Including the linear system that the GMRES(50) solver did not converge, this solver preconditioned by the ARMS preconditioner did not converge when applied to the instance *tmt_unsym*, whose simulations terminated for exceeding the timeout stipulated of 20 min.

The GMRES(50) solver preconditioned by the ILUT preconditioner converged to five out of nine linear systems used in this study. The GMRES(50) solver preconditioned by the ILUC preconditioner converged to only one linear system.

6 Conclusions

This paper applied 13 low-cost symmetric permutations to reduce the computing time of the preconditioned GMRES(50) solver. Consistent with the current findings, the experiments conducted in this computational experiment show that the choice of the best combination of reordering algorithm and preconditioned GMRES solver is strongly problem-dependent. This paper indicated promising combinations of preconditioners and reordering heuristics (or the use of the original ordering of the instance) for fast convergence of a GMRES(50) method in nine large-scale linear systems.

A suitable parameter m for the restarted GMRES solver could be very different for each matrix analyzed. Similar to several publications in this field, however, and since the objective was to evaluate the influence of reordering algorithms on the convergence of the restarted GMRES solver, after exploratory investigations (see Sect. 4), we fixed the number of restart vectors for the GMRES solver for all matrices used.

In experiments using nine linear systems (with sizes ranging from 411,676 to 917,825 unknowns and from 1,876,011 to 4,584,801 non-null coefficients) comprised of real nonsymmetric matrices contained in the SuiteSparse matrix collection [22], the RBFS [4] (in three instances), BFS (in one instance), and Reverse Cuthill-McKee [21] (in one instance) orderings yielded the best results to reduce the processing times of the preconditioned GMRES(50) solver. In particular,

the George-Liu algorithm [23] did not favor the reordering algorithms in the experiments performed in this study. The experiments included four problems (*largebasis*, *rajat29*, *rajat30*, and *ASIC 680k*) in which the GMRES(50) solver preconditioned by a specific preconditioner converged much faster after a permutation of the coefficient matrix (see Table 6) than using the original ordering of the instance.

Previous works in this field (e.g., [2,4,5]) found that the natural ordering rarely provides the best results, and was usually the worst. The experiments conducted in this study also evaluated the reordering algorithms when applied to the original sequence of unknowns and equations given in the linear systems. The results showed that reordering the matrices did not favor the computing time of the GMRES(50) solver (at least with the parameters used) when applied to three out of nine large-scale test matrices used. Additionally, the computing time of the preconditioned GMRES(50) solver applied to the original ordering yielded similar times to the best results achieved when reordering the rows and columns of another instance (*tmt_unsym*). Some characteristics can explain these different results from previous publications in this subject [2,4,5]. First, the experiments performed in the present study employed different preconditioners. We used the ILUC and ARMS [9] preconditioners, including the ILUT algorithm. Second, the size of the dropped entries strongly influences the effectiveness of incomplete factorizations. The choice of parameters of the GMRES method in tandem with each preconditioner was also conducted quite differently in this computational experiment from those previous publications. Subtle differences in the parameter values of incomplete factorization preconditioners are determinants for the convergence of the GMRES solver. Further analysis of the parameters of the preconditioners could show better results. However, the experiments conducted in the present study showed that the choice of the best parameters is not trivial. Third, the instances included in the experiments are much larger than those used by those researchers. Fourth, the proper set of instances can explain these results, since the best combination of reordering algorithm and preconditioned GMRES solver is highly problem-dependent, as previously mentioned. Fifth, similar to those previous publications, we limited the experiments to symmetric permutations, and we used nonsymmetric linear systems.

The results of the experiments showed that the heuristics for profile reduction used are not recommended (i.e., Sloan's algorithm [17] and its variants, NSloan [18], Sloan-MGPS [19], and MPG [16] heuristics). Moreover, heuristics that yield further profile reductions than Sloan's algorithm [17] and its variants at higher execution costs may not favor the computing time of the preconditioned restarted GMRES method when applied to large-scale linear systems.

One can employ the BFS or RCM procedure or a variant (e.g., a reverse BFS [4]) for preprocessing matrices for the GMRES(50) solver. The RBFS ordering yielded, in general, better performance than the classical RCM ordering [21]. Furthermore, the experiments conducted in the present study showed that the

use of the George-Liu algorithm [23] did not favor the convergence of the preconditioned GMRES(50) solver.

We plan to investigate the effects of parallel orderings on the convergence of parallel implementations of the restarted preconditioned GMRES solver using OpenMP, Galois, and Message Passing Interface systems. The literature on heuristics for bandwidth reduction presents many works with sequential algorithms and just a small fraction of parallel strategies. A systematic review of parallel heuristics for bandwidth reduction is another future step in our study. Concerning massively parallel computing, we plan to evaluate heuristics for bandwidth reduction implemented within the Intel® Math Kernel Library running on Intel® Scalable processors.

References

1. Gonzaga de Oliveira, S.L., Carvalho, C., Osthoff, C.: The effect of symmetric permutations on the convergence of are started GMRES solver with ILU-type preconditioners. In: 2019 Winter Simulation Conference (WSC), National Harbor, MA,USA, pp. 3219–3230. IEEE (2019)
2. Benzi, M., Haws, J.C., Tuma, M.: Preconditioning highly indefinite and nonsymmetric matrices. SIAM J. Sci. Comput. **22**(4), 1333–1353 (2000)
3. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. **7**, 856–869 (1986)
4. Benzi, M., Szyld, D.B., Van Duin, A.: Orderings for incomplete factorization preconditioning of nonsymmetric problems. SIAM J. Sci. Comput. **20**(5), 1652–1670 (1999)
5. Camata, J.J., Rossa, A.L., Valli, A.M.P., Catabriga, L., Carey, G.F., Coutinho, A.L.G.A.: Reordering and incomplete preconditioning in serial and parallel adaptive mesh refinement and coarsening flow solutions. Int. J. Numer. Meth. Fl. **69**(4), 802–823 (2012)
6. Gonzaga de Oliveira, S.L., Bernardes, J.A.B., Chagas, G.O.: An evaluation of reordering algorithms to reduce the computational cost of the incomplete Cholesky-conjugate gradient method. Comput. Appl. Math. **37**(3), 2965–3004 (2017). https://doi.org/10.1007/s40314-017-0490-5
7. Saad, Y.: ILUT: a dual threshold incomplete LU factorization. Numer. Linear Algebra Appl. **1**(4), 387–402 (1994)
8. Li, N., Saad, Y., Chow, E.: Crout versions of ILU for general sparse matrices. SIAM J. Sci. Comput. **25**(2), 716–728 (2003)
9. Saad, Y., Suchomel, B.: ARMS: an algebraic recursive multilevel solver for general sparse linear systems. Numer. Linear Algebra Appl. **9**(5), 359–378 (2002)
10. Saad, Y.: ITSOL vol 2.0: Iterative solvers package (2016)
11. Gonzaga de Oliveira, S.L., Bernardes, J.A.B., Chagas, G.O.: An evaluation of low-cost heuristics for matrix bandwidth and profile reductions. Comput. Appl. Math. **37**(2), 1412–1471 (2016). https://doi.org/10.1007/s40314-016-0394-9
12. George, A., Liu, J.W.: Computer Solution of Large Sparse Positive Definite Systems. Prentice-Hall, Englewood Cliffs (1981)
13. Gonzaga de Oliveira, S.L., Abreu, A.A.A.M., Robaina, D.T., Kischnhevsky, M.: An evaluation of four reordering algorithms to reduce the computational cost of the Jacobi-preconditioned conjugate gradient method using high-precision arithmetic. Int. J. Bus. Intell. Data Min. **12**(2), 190–209 (2017)

14. Gonzaga de Oliveira, S.L., Abreu, A.A.A.M.: An evaluation of pseudoperipheral vertex finders for the reverse Cuthill-Mckee method for bandwidth and profile reductions of symmetric matrices. In: Proceedings of the 37th International Conference of the Chilean Computer Science Society (SCCC), Santiago, Chile, pp. 1–9. IEEE (2018). https://doi.org/10.1109/SCCC.2018.8705263
15. Koohestani, B., Poli, R.: A hyper-heuristic approach to evolving algorithms for bandwidth reduction based on genetic programming. In: Bramer, M., Petridis, M., Nolle, L. (eds.) Research and Development in Intelligent Systems XXVIII, SGAI 2011, pp. 93–106. Springer, London (2011). https://doi.org/10.1007/978-1-4471-2318-7_7
16. Medeiros, S.R.P., Pimenta, P.M., Goldenberg, P.: Algorithm for profile and wavefront reduction of sparse matrices with a symmetric structure. Eng. Comput. **10**(3), 257–266 (1993)
17. Sloan, S.W.: A Fortran program for profile and wavefront reduction. Int. J. Numer. Meth. Eng. **28**(11), 2651–2679 (1989)
18. Kumfert, G., Pothen, A.: Two improved algorithms for envelope and wavefront reduction. BIT Numer. Math. **37**(3), 559–590 (1997)
19. Reid, J.K., Scott, J.A.: Ordering symmetric sparse matrices for small profile and wavefront. Int. J. Numer. Meth. Eng. **45**(12), 1737–1755 (1999)
20. Cuthill, E., McKee, J.: Reducing the bandwidth of sparse symmetric matrices. In: Proceedings the 1969 24th International Conference, pp. 157–172. ACM (1969)
21. George, A.: Computer implementation of the finite element method. PhD thesis, Stanford University, Stanford (1971)
22. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 1–25 (2011)
23. George, A., Liu, J.W.H.: An implementation of a pseudoperipheral node finder. ACM Trans. Math. Softw. **5**(3), 284–295 (1979)