Robustness and Accuracy in Pipelined Bi-Conjugate Gradient Stabilized Methods

Mykhailo Havdiak[1], José I. Aliaga[2], and Roman Iakymchuk[3]

[1] Ivan Franko National University of Lviv, Ukraine
mykhailo.havdiak@lnu.edu.ua
[2] Universitat Jaime I, Spain
aliaga@icc.uji.es
[3] Umeå University and Uppsala University, Sweden
riakymch@cs.umu.se

Abstract. In this article, we propose an accuracy-assuring technique for finding a solution for unsymmetric linear systems. Such problems are related to different areas such as image processing, computer vision, and computational fluid dynamics. Parallel implementation of Krylov subspace methods speeds up finding approximate solutions for linear systems. In this context, the refined approach in pipelined BiCGStab enhances scalability on distributed memory machines, yielding to substantial speed improvements compared to the standard BiCGStab method. However, it's worth noting that the pipelined BiCGStab algorithm sacrifices some accuracy, which is stabilized with the residual replacement technique. This paper aims to address this issue by employing the ExBLAS-based reproducible approach. We validate the idea on a set of matrices from the SuiteSparse Matrix Collection.

Keywords: Krylov subspace methods; BiCGStab; Residual replacement; Numerical reliability; ExBLAS; HPC.

1 Introduction

Krylov subspace methods form a powerful class of iterative techniques for solving large linear systems arising in diverse scientific and engineering applications. These methods are particularly well-suited for problems where the coefficient matrix is sparse and both symmetric or non-symmetric. Such methods are applicable and often used in image denoising, data compression, inverse problems, and other areas. The Conjugate Gradient (CG) method, introduced in [6], is one of the earliest members of this well-known class of iterative solvers. However, CG is limited to solving symmetric and positive definite (SPD) systems. In contrast, the Bi-Conjugate Gradient [3] (BiCG) method extends its applicability to more general classes of non-symmetric and indefinite linear systems. Additionally, the Conjugate Gradient Squared [8] (CGS) method provides an alternative approach. The BiConjugate Gradient Stabilized (BiCGStab) method [9] was introduced as a smoother converging version of both BiCG and CGS methods. Preconditioning is usually incorporated in real implementations of these methods in order to accelerate the convergence of the methods and improve their numerical features.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

These classical Krylov subspace methods have been actively discussed and optimized. For instance, optimizations have been explored for a specific class of hepta-diagonal sparse matrices on GPUs, as well as the implementation of the pipelined Bi-Conjugate Gradient Stabilized method (p-BiCGStab) [1] to overlap (hide) communication and computation. The pipelined methods, in particular, introduced more operations compared to the original ones and with that impacted the convergence as the computer residual deviated from the true one. As a remedy, the residual replacement technique was proposed [1] to numerically stabilize convergence with a strong emphasis on mathematical aspects.

The purpose of this initial study is to explore the possibility of avoiding residual replacement in pipelined Krylov-type methods [1] with the help of accurate and reproducible computations via the ExBLAS approach [4,5]. As a test case, we use the pipelined BiCGStab method.

2 Reproducibility of BiCGStab and pipelined BiCGStab

BiCGStab was developed to solve non-symmetric linear systems while avoiding the often irregular convergence patterns of the CGS method. In BiCGStab, minimizing a residual vector promotes smoother convergence. However, when the Generalized Minimal Residual method (GMRES) [7] stagnates, preventing the expansion of the Krylov subspace, BiCGStab may fail to proceed effectively.

In the light of the conventional BiCGStab algorithm, see Alg. 1, introduced by Van der Vorst, Cools and Vanroose proposed an optimization known as pipelined BiCGStab (p-BiCGStab) [1], see Alg. 2. This optimization entails two primary phases within the pipelining framework. Firstly, in what is termed the 'communication-avoiding' phase, the standard Krylov algorithm undergoes a transformation into a mathematically equivalent form with the reduced global synchronization points. This reduction is accomplished by merging the global reduction phases of various dot products scattered throughout the algorithm into a single global communication phase. Subsequently, in the 'communication-hiding' phase, the algorithm is further refined to overlap the remaining global reduction phases with the sparse matrix-vector product and application of the preconditioner. This strategic restructuring effectively mitigates the typical communication bottleneck by concealing communication time behind productive computational tasks. Although, the methods are mathematically equivalent they may lead to different numerical results and convergence patterns due to the non-associativity of floating-point operations.

The BiCGStab (Alg. 1) and p-BiCGStab (Alg. 2) methods will serve as the primary methods utilized throughout this article, although we mainly focus on the p-BiCGStab. Due to the non-associativity of finite precision floating-point operations, the mathematical equivalent of these two methods can show large divergence while implemented in parallel environments especially for the tolerance 10^{-6} and below. To stabilize this deviation in p-BiCGStab, the residual replacement technique was proposed [1]. This technique resets the residuals r_i, along with the auxiliary variables w_i, s_i, and z_i, to their original values every k iterations. In the preconditioned version, this process also updates $\bar{r_i} = M^{-1} r_i$ and $\bar{s_i} = M^{-1} s_i$, where M is the preconditioning operator. We refer to [1] for more details.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

Algorithm 1 BiCGStab [9]

function BiCGStab(A, b, x_0)
  $r_0 := b - Ax_0$
  $p_0 := r_0$
  for i = 0, 1, 2, ... **do**
    $s_i := Ap_i$
    $a_i := (r_0, r_i)/(r_0, s_i)$
    $q_i := r_i - a_i s_i$
    $y_i := Aq_i$
    $w_i := (q_i, y_i)/(y_i, y_i)$
    $x_{i+1} := x_i + a_i p_i + w_i q_i$
    $r_{i+1} := q_i - w_i y_i$
    $\beta_i := (a_i/w_i)(r_0, r_{i+1})/(r_0, r_i)$
    $p_{i+1} := r_{i+1} + \beta_i(p_i - w_i s_i)$
  end for
end function

Algorithm 2 Pipelined BiCGStab [1]

function p-BiCGStab(A, b, x_0)
  $r_0 := b - Ax_0$
  $w_0 := A - r_0$
  $t_0 := Aw_0$
  $a_0 := (r_0, r_0)/(r_0, w_0)$
  $\beta_{-1} := 0$
  for i = 0, 1, 2, ... **do**
    $p_i := r_i + \beta_{i-1}(p_{i-1} - w_{i-1}s_{i-1})$
    $s_i := w_i + \beta_{i-1}(s_{i-1} - w_{i-1}z_{i-1})$
    $z_i := t_i + \beta_{i-1}(z_{i-1} - w_{i-1}v_{i-1})$
    $q_i := r_i - a_i s_i; \quad y_i := w_i - a_i z_i$
    $v_i := Az_i; \quad w_i := (q_i, y_i)/(y_i, y_i)$
    $x_{i+1} := x_i + a_i p_i + w_i q_i$
    $r_{i+1} := q_i - w_i y_i$
    $w_{i+1} := y_i - w_i(t_i - a_i v_i); \quad t_{i+1} := Aw_{i+1}$
    $\beta_i := (a_i/w_i)(r_0, r_{i+1})/(r_0, r_i)$
    $a_{i+1} := (r_0, r_{i+1})/((r_0, w_{i+1}) + \beta_i(r_0, s_i) - \beta_i w_i(r_0, z_i))$
  end for
end function

In [5], we proposed to ensure the reproducibility and accuracy of the pure MPI implementation of the preconditioned BiCGStab method via the ExBLAS approach. ExBLAS combines together long accumulator and floating-point expansions into algorithmic solutions as well as efficiently tunes and implements them on various architectures. ExBLAS aims to provide new algorithms and implementations for fundamental linear algebra operations (like those included in the BLAS library), that deliver reproducible and accurate results with small or without losses to their performance on modern parallel architectures such as desktop and server processors, Intel Xeon Phi co-processors, and GPU accelerators. We construct our approach in such a way that it is independent of data partitioning, order of computations, thread scheduling, or reduction tree schemes. Instead of using the residual replacement technique, we propose to exhibit the benefits of the ExBLAS approach in the pipelined BiCGStab method.

3 Experimental Results

This section presents a series of numerical experiments to evaluate the convergence, performance, and accuracy of the BiCGStab methods, including the reproducible with ExBLAS. The results include comparisons between BiCGStab, pipelined BiCGStab (p-BiCGStab), p-BiCGStab with ExBLAS, and p-BiCGStab with the residual replacement technique across various matrices from the Suite Sparse Matrix Collection [2]. In the experiments, IEEE754 double-precision arithmetic was utilized, and we run on nodes at HPC2N with the dual 14-core Intel Xeon Gold 6132 (Skylake) @2.60GHz interconnected via EDR Infiniband.

The SuiteSparse Matrix Collection is a comprehensive repository of sparse matrices widely used for benchmarking and testing numerical algorithms in the field of computational mathematics. It allows for robust and repeatable experiments, as performance results with artificially generated matrices can be mis-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

leading. Hence, repeatable experiments are crucial for ensuring the reliability of algorithm evaluations. The collection encompasses a diverse range of matrices representing real-world problems from various disciplines.

Tab. 1 presents the comparative performance of four iterative BiCGStab methods, namely BiCGStab, pipelined BiCGStab (p-BiCGStab), p-BiCGStab with ExBLAS, and pipelined BiCGStab with residual replacement (p-BiCGStab-RR), across a selection of sparse matrices from the SuiteSparse Matrix Collection. Each cell in the table represents the number of iterations required to achieve convergence for a specific method and a given matrix, with convergence thresholds set at 10^{-6} and 10^{-9}. The results demonstrate varying convergence behavior among the methods across different matrices, providing insights into their respective efficiency in solving sparse linear systems.

Problem	BiCGStab		p-BiCGStab		p-BiCGStabExBLAS		p-BiCGStabRR	
	10^{-6}	10^{-9}	10^{-6}	10^{-9}	10^{-6}	10^{-9}	10^{-6}	10^{-9}
1138_bus	30	151	**27**	130	35	**108**	**27**	130
add32	**38**	74	**38**	**68**	38	**69**	38	**68**
bcsstk13	545	-	520	2258	350	3273	**195**	**403**
bcsstk14	149	461	44	459	**43**	**433**	44	-
bcsstk18	405	2806	**261**	1284	366	**1274**	309	-
bcsstk27	283	**958**	335	2107	**279**	1477	335	2107
bfwa782	99	647	74	**448**	**54**	463	115	576
cdde6	36	122	**34**	121	**34**	**115**	**34**	388
msc01050	**29**	61	30	**47**	**28**	60	30	**47**
msc04515	123	**257**	**96**	275	**98**	308	263	334
orsreg_1	**21**	161	22	168	**20**	**106**	22	371
pde2961	**100**	**166**	111	278	134	287	170	683
plat1919	**79**	**132**	99	185	84	179	87	250
rdb3200l	31	193	31	223	31	**171**	31	567
saylr4	28	73	28	74	28	**69**	28	74
sherman3	34	501	33	314	**26**	400	33	**271**
utm5940	99	592	97	**420**	**18**	603	20	**419**

Table 1: Number of iterations for the BiCGStab-like methods on a set of the SuiteSparse matrices without precondition. The initial estimate is a zero vector x_0. The best-performing method is highlighted in bold.

When summarizing the findings, several notable observations come to light. Firstly, for $\varepsilon = 10^{-6}$, all methods demonstrate comparable performance in numerous cases. However, when considering a higher tolerance, $\varepsilon = 10^{-9}$, the p-BiCGStabExBLAS method consistently outperforms p-BiCGStab across the majority of matrices, owing to its enhanced accuracy. Moreover, the p-BiCGStab with residual replacement strategy (p-BiCGStabRR) method generally exhibits superior convergence rates compared to the p-BiCGStab method. Yet, increasing the tolerance also reveals instances where the classic BiCGStab method proves more efficient in terms of iterations, although it may encounter convergence issues for certain matrices. Notably, for the specific `bcsstk13` problem, p-BiCGStabRR demonstrates the most favorable convergence characteristics for both $tol = 10^{-6}$, $tol = 10^{-9}$. Despite these advantages, there are scenarios where

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

p-BiCGStabRR fails to converge, namely `bcsstk14` and `bcsstk18`. The accuracy of the p-BiCGStabRR is highly contingent to the specific problem context and parameter choices, leading to variability in its effectiveness. This method may exhibit convergence speed under certain conditions while performing poorly under others, highlighting the sensitivity of its outcomes to these factors. The method requires multiple runs to determine the optimal step and the best place for applying residual replacement. A less optimal parameter choice can result in more iterations compared to the pipelined-BiCGStab method.

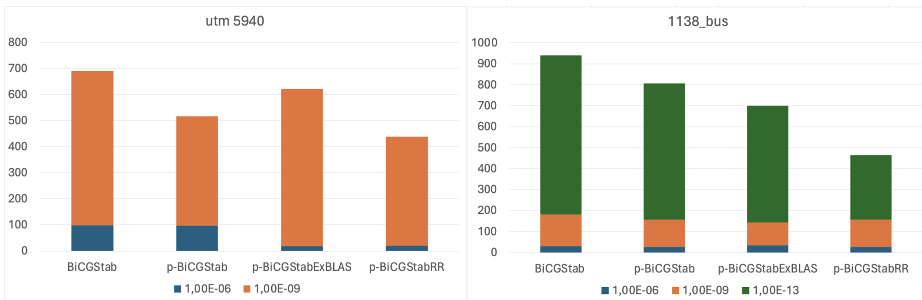

Fig. 1: Number of iterations required by various BiCGStab-like methods to achieve a specified tolerance $(10^{-6}, 10^{-9}, 10^{-13})$. p-BiCGStabRR stands for the pipelined version of the BiCGStab method with residual replacement; P-BiCGStabExBLAS refers to the method with ExBLAS.

In Fig. 1, the `utm5940` case highlights an interesting trend: the ExBLAS version performed the best for epsilon 10^{-6}, yet with an increase to 10^{-9}, it required slightly more iterations compared to other methods. Overall, p-BiCGStabExBLAS demonstrates good constant performance in terms of iterations. When examining p-BiCGStabRR, it's evident that for certain examples, it exhibits the lowest iteration count. However, there are instances for higher tolerance the method results in significantly higher iteration counts compared to other methods.

Fig. 2 provides the convergence history of the four BiCGStab-like methods. We can observe the performance characteristics of the methods on a particular problem instance, where the tolerance is set to 10^{-13}. The pipelined BiCGStab method with ExBLAS consistently outperforms the regular pipelined variant in terms of iterations across a wide range of scenarios. p-BiCGStabExBLAS exhibits a reduced occurrence of spikes compared to p-BiCGStabRR, suggesting a smoother and more stable performance profile. This difference highlights the potential of ExBLAS to offer improved reliability and predictability in computational outcomes, namely results and iterations.

Following this, we evaluate the four considered methods using an increased number of processes. Subsequently, we present Tabs. 2 and 3 illustrating the outcomes achieved by the aforementioned methods across different process counts. A notable observation from both Tab. 2 and Tab. 3 is the consistency in the number of iterations required for the ExBLAS implementation across different numbers of processes. Thus, increasing the number of processes does not lead to

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

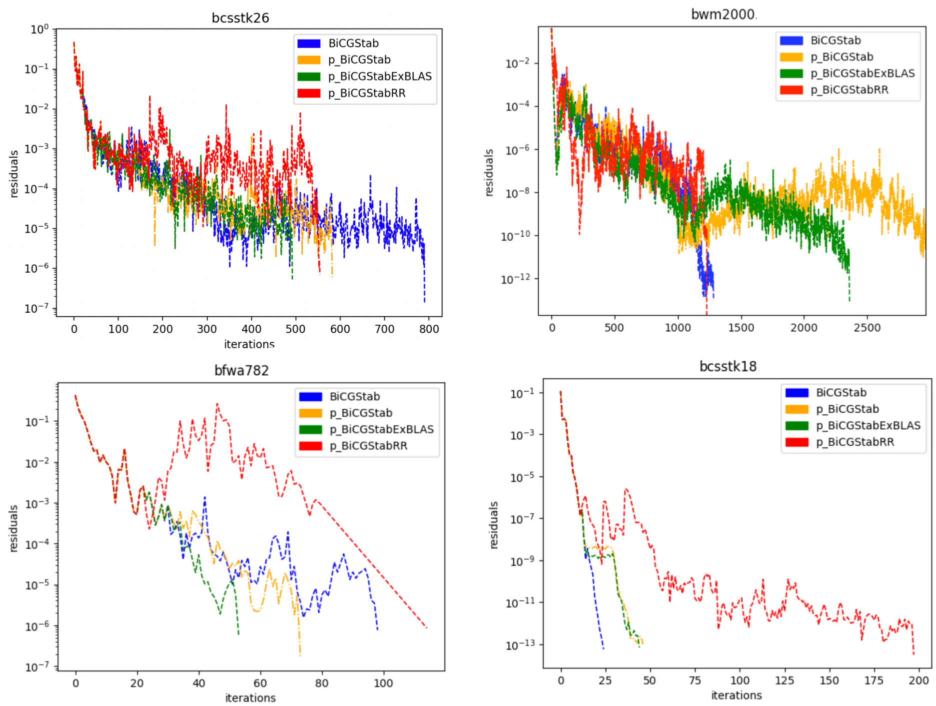

Fig. 2: Residual history of the four BiCGStab-like methods; $tol = 10^{-13}$.

$\varepsilon = 10^{-6}$	BiCGStab			p-BiCGStab			p-BiCGStabExBLAS		
	$n01$	$n08$	$n16$	$n01$	$n08$	$n16$	$n01$	$n08$	$n16$
bcsstk26	791	235	599	583	528	683	493	493	493
bwm2000	37	37	37	37	37	37	37	37	37
bfwa782	99	79	85	74	104	59	54	54	54
bcsstk18	405	414	363	261	305	356	366	366	366

Table 2: Number of iterations required for BiCGStab, p-BiCGStab, and p-BiCGStabExBLAS by varying numbers of processes (nXX) for $\varepsilon = 10^{-6}$.

a faster solution for pipelined BiCGStab method. The pipelined BiCGStabRR as indicated in Tab. 1 demonstrates its best outcome of 195 iterations for the `bcsstk13` matrix.

Tab. 4 illustrates the iteration counts for the BiCGStab, p-BiCGStab, and p-BiCGStabExBLAS methods across different numbers of processes. The method with residual replacement technique finds an approximation to the solution only on a single process. Additionally, the table presents the execution times for each scenario. p-BiCGStabExBLAS requires more time, attributed to its higher accuracy. Nonetheless, the overhead associated with p-BiCGStabExBLAS diminishes as the number of processes increases, dropping from 2.6x on a single process to 1.87x on 16 processes.

Fig. 3 illustrates the benefits of using pipelined methods within a parallel environment, emphasizing their efficiency and scalability. Certainly, the scale is small but the gain starts to be visible on 16 processes, four per each of four

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

Matrix	BiCGStab			p-BiCGStab			p-BiCGStabExBLAS		
	$n01$	$n08$	$n16$	$n01$	$n08$	$n16$	$n01$	$n08$	$n16$
bwm2000	1268	1267	1120	663	1131	1024	1232	1232	1232
bfwa782	647	528	531	448	476	498	464	463	463
bcsstk18	2806	2819	2286	1284	1811	2318	1274	1274	1274

Table 3: Number of iterations required for BiCGStab, p-BiCGStab, and p-BiCGStabExBLAS by varying the numbers of processes (nXX) for $tol = 10^{-9}$.

Method	$n01$		$n08$		$n16$	
	iter	time	iter	time	iter	time
BiCGStab	545	2.034×10^{-1}	520	4.8236×10^{-2}	544	3.94×10^{-2}
p-BiCGStab	520	2.0495×10^{-1}	482	4.457×10^{-2}	394	3.106×10^{-2}
p-BiCGStabE	350	5.321×10^{-1}	350	8.976×10^{-2}	350	5.819×10^{-2}
p-BiCGStabRR	195	6.198×10^{-2}	-	-	-	-

Table 4: Number of iterations and time required for the BiCGStab-like methods

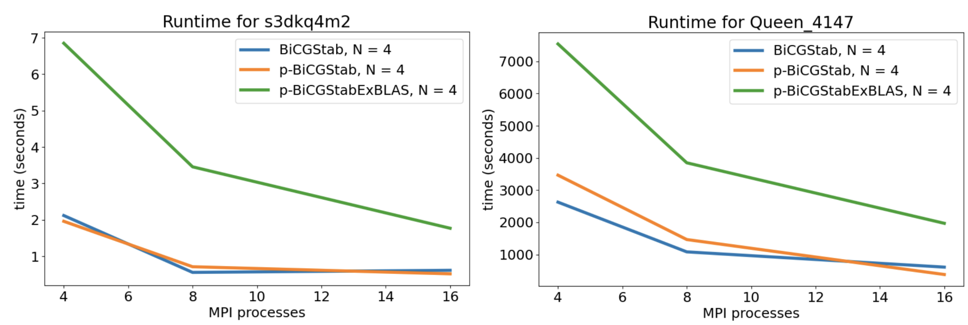

Fig. 3: Runtime comparison of BiCGStab-like methods on two matrices: s3dkq4m2 with $4,427,725$ nnz $tol = 10^{-9}$; Queen_4147 with $316,548,962$ nnz and $tol = 10^{-6}$. We used four nodes with $1, 2$, and 4 MPI processes each.

nodes; we used only few processes per node to highlight the benefit. In this test, we focus on two large matrices: `s3dkq4m2` with $4,427,725$ non-zero elements and `Queen_4147` with $316,548,962$ non-zero elements. Larger-dimensional problems tend to demonstrate better strong scalability in parallel environments, using the existing potential of available resources especially on four and eight processes. Conversely, employing 16 processes for the `s3dkq4m2` matrix with fewer non-zero elements did not yield significant improvements for BiCGStab and p-BiCGStab. Additionally, p-BiCGStabExBLAS demonstrates strong scalability for both problems due to more flops imposed by the ExBLAS approach. Overhead for s3dkq4m2 varied from 3.41x to 4.8x, and matrix Queen_4147 from 1.96x to 5.2x. With the increase in the number of processes from 4 to 16, the execution time for p-BiCGStabExBLAS was reduced by more than 2x.

4 Conclusion

In this study, we investigated the robustness and accuracy of the pipelined Biconjugate Gradient Stabilized using the ExBLAS approach as not only an accurate and reproducible solution but also as an alternative to the residual replacement technique. Our analysis focused on evaluating the convergence be-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_35

havior of the method across a set of matrices from the SuiteSparse Matrix Collection. Through the numerical experiments, we demonstrated that the pipelined BiCGStab method with ExbLAS, consistently outperforms the conventional pipelined BiCGStab approach in terms of convergence rates and numerical reliability. Overall, this study emphasizes the importance of considering algorithmic refinements and numerical stability enhancements to achieve reliable and efficient solutions for challenging computational problems. The results underscore notable performance disparities among the assessed methods. Specifically, BiCGStab demonstrates better stability compared to p-BiCGStab, showcasing its reliability in solving linear systems. The residual replacement strategy is expected to address the stability of the pipelined method, bringing it closer to the robustness exhibited by BiCGStab. Although its performance is inferior to ExBLAS implementation. This suggests that the ExBLAS implementation capitalizes on the advantages of the pipelined method version while maintaining stability as in BiCGStab.

As a future work, we shall conduct theoretical study of the ExBLAS approach as a possible replacement for the residual replacement, which requires some empirical trials to get the right step. Furthermore, we plan to carry out an exhaustive study with more matrices from the SuiteSparse Matrix Collection as well as the real applications like the ones from the EU-funded EuroHPC JU Center of Excellence in Exascale CFD (CEEC)[4], where the last author contributes.

References

1. Cools, S., Vanroose, W.: The communication-hiding pipelined bicgstab method for the parallel solution of large unsymmetric linear systems. ParCo **65**, 1–20 (2017). https://doi.org/10.1016/j.parco.2017.04.005
2. Davis, T.A., Hu, Y.: The university of florida sparse matrix collection. ACM TOMS **38**(1) (2011). https://doi.org/10.1145/2049662.2049663
3. Fletcher, R.: Conjugate gradient methods for indefinite systems. Springer LNM pp. 73–89 (1976). https://doi.org/10.1007/BFb0080116
4. Iakymchuk, R., Collange, S., Defour, D., Graillat, S.: ExBLAS: Reproducible and accurate BLAS library. In: NRE at SC15 (2015), https://hal.science/hal-01202396v3
5. Iakymchuk, R., Graillat, S., Aliaga, J.I.: General framework for re-assuring numerical reliability in parallel krylov solvers: A case of bicgstab methods. IJHPCA **38**, 17–33 (2024). https://doi.org/10.1177/10943420231207642
6. M.R. Hestenes, E.S.: Methods of conjugate gradients for solving linear systems. NBS **49** (1952). https://doi.org/10.6028/jres.049.044
7. Saad, Y., Schultz, M.H.: Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comp **7**, 856–869 (1986). https://doi.org/10.1137/0907058
8. Sonneveld, P.: Cgs, a fast lanczos-type solver for nonsymmetric linear systems. SIAM J Sci Stat Comp **10**(1), 36–52 (1989). https://doi.org/10.1137/0910004
9. van der Vorst, H.A.: Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. SIAM J Sci Stat Comp **13**(2), 631–644 (1992). https://doi.org/10.1137/0913035

[4] This work was partially supported by CEEC under grant agreement No 10109339