Analyzing the Effect of Local Rounding Error Propagation on the Maximal Attainable Accuracy of the Pipelined Conjugate Gradient Method

Siegfried Cools, Emrullah Fatih Yetkin, Emmanuel Agullo, Luc Giraud, Wim Vanroose

SIAM J. MATRIX ANAL. APPL.
Vol. 39, No. 1, pp. 426–450

ANALYZING THE EFFECT OF LOCAL ROUNDING ERROR PROPAGATION ON THE MAXIMAL ATTAINABLE ACCURACY OF THE PIPELINED CONJUGATE GRADIENT METHOD*

SIEGFRIED COOLS†, EMRULLAH FATIH YETKIN‡, EMMANUEL AGULLO§, LUC GIRAUD§, AND WIM VANROOSE†

Abstract. Pipelined Krylov subspace methods typically offer improved strong scaling on parallel HPC hardware compared to standard Krylov subspace methods for large and sparse linear systems. In pipelined methods the traditional synchronization bottleneck is mitigated by overlapping time-consuming global communications with useful computations. However, to achieve this communication-hiding strategy, pipelined methods introduce additional recurrence relations for a number of auxiliary variables that are required to update the approximate solution. This paper aims at studying the influence of local rounding errors that are introduced by the additional recurrences in the pipelined Conjugate Gradient (CG) method. Specifically, we analyze the impact of local round-off effects on the attainable accuracy of the pipelined CG algorithm and compare it to the traditional CG method. Furthermore, we estimate the gap between the true residual and the recursively computed residual used in the algorithm. Based on this estimate we suggest an automated residual replacement strategy to reduce the loss of attainable accuracy on the final iterative solution. The resulting pipelined CG method with residual replacement improves the maximal attainable accuracy of pipelined CG while maintaining the efficient parallel performance of the pipelined method. This conclusion is substantiated by numerical results for a variety of benchmark problems.

Key words. conjugate gradients, parallelization, latency hiding, global communication, pipelining, rounding errors, maximal attainable accuracy

AMS subject classifications. 65F10, 65F50, 65G50, 65Y05, 65Y20

DOI. 10.1137/17M1117872

1. Introduction.

Krylov subspace methods [25, 31, 32, 39, 48] form the basis linear algebra solvers for many contemporary high-performance computing applications. The Conjugate Gradient (CG) method, which dates back to the 1952 paper by Hestenes and Stiefel [29], can be considered as the first of these methods. Although over 60 years old, the CG method is still the workhorse method for the solution of linear systems with symmetric positive definite (SPD) matrices due to its numerical simplicity and easy implementation. These SPD systems may originate from various applications, such as the discretization of PDEs.

Due to the transition of hardware towards the exascale regime in the coming years, research on the scalability of Krylov subspace methods on massively parallel architectures has become increasingly prominent [14, 15]. This is reflected in the new High Performance Conjugate Gradients benchmark for ranking HPC systems introduced

*Received by the editors February 22, 2017; accepted for publication (in revised form) November 27, 2017; published electronically March 13, 2018.
http://www.siam.org/journals/simax/39-1/M111787.html
Funding: The work of the authors was funded by the EXA2CT European Project on Exascale Algorithms and Advanced Computational Techniques, which receives funding from the EU's Seventh Framework Programme (FP7/2007-2013) under grant agreement 610741. Additionally, the work of the first author was funded by the Research Foundation Flanders (FWO) under grant 12H4617N.

†Applied Mathematics Group, Department of Mathematics and Computer Science, University of Antwerp, 2020 Antwerp, Belgium (siegfried.cools@uantwerp.be, wim.vanroose@uantwerpen.be).
‡Department of Computer Engineering, Istanbul Kemerburgaz University, 34217 Bağcılar, Turkey (fatih.yetkin@kemerburgaz.edu.tr).
§HiePACS, Inria Bordeaux, Sud-Ouest, 33405 Talence, France (emmanuel.agullo@inria.fr, luc.giraud@inria.fr).

by Dongarra et al. in 2013 [16, 17]. The ranking is based on sparse matrix-vector computations and data access patterns, rather than the dense matrix algebra used in the traditional High Performance LINPACK benchmark. Moreover, since the system matrix is often sparse, the main bottleneck for efficient parallel execution is typically not the sparse matrix-vector product (SPMV) but the communication overhead (bandwidth saturation) caused by global reductions required in the computation of dot-products.

Recently, significant research has been devoted to the mitigation and/or elimination of the synchronization bottleneck in Krylov subspace methods. The earliest papers on synchronization reduction and latency hiding in Krylov subspace methods date back to the late 1980s [44] and early 1990s [2, 9, 10, 13, 19]. The idea of reducing the number of global communication points in Krylov subspace methods on parallel computer architectures was also used in the s-step methods by Chronopoulos et al. [6, 7, 8] and more recently by Carson et al. in [3, 4]. In addition to communication-avoiding methods,[1] research on hiding global communication by overlapping communication with computations was performed by a various authors over the last decades; see Demmel et al. [13], De Sturler et al. [11], and Ghysels et al. [21, 22]. We refer the reader to the recent work [5, section 2] and the references therein for more background and a wider historical perspective on the development of early variants of the CG algorithm that contributed to the current algorithmic strive towards parallel efficiency.

The pipelined CG (p-CG) method proposed in [22] aims at hiding the global synchronization latency of standard preconditioned CG by removing some of the global synchronization points. Pipelined CG performs only one global reduction per iteration. Furthermore, this global communication phase is overlapped by the SPMV, which requires only local communication. In this way, idle core time is minimized by performing useful computations simultaneously to the time-consuming global communication phase; cf. [18].

The reorganization of the CG algorithm that is performed to achieve the overlap of communication with computations introduces several additional AXPY ($y \leftarrow \alpha x + y$) operations to recursively compute auxiliary variables. Vector operations such as an AXPY are typically computed locally and thus do not require communication between nodes. Thus, the addition of extra recurrences has no impact on the communication flow of the algorithm. Dot-products of two vectors, on the other hand, involve global communication between all processes and are therefore grouped together in p-CG.

In exact arithmetic, the resulting pipelined CG algorithm is equivalent to classical CG. However, when switching to finite precision, each of the additional recurrences introduces local rounding errors. The propagation of these rounding errors throughout the algorithm is much more pronounced for pipelined CG and can have a detrimental effect on the iterative solution. As a result, a significant loss of attainable accuracy compared to classical CG can in practice be observed for the p-CG algorithm. The current paper contributes to the analysis of the rounding error propagation in different variants of the CG algorithm. Additionally, the analytical results will be used to formulate an automated residual replacement strategy that improves the maximal attainable accuracy of the pipelined CG method. We stress that the proposed residual

[1] Although commonly used in the contemporary literature, the term "communication-avoiding" Krylov subspace algorithm is slightly dubious since the number of global synchronization phases is in fact *reduced* by reformulating the algorithms rather than *avoided*; hence, the term "communication-reducing" algorithm may be more appropriate in this context.

replacement strategy only accounts for an improvement of the attainable accuracy in pipelined CG. Other notable rounding error effects in multiterm recurrence algorithms, such as a delay of convergence due to loss of orthogonality in finite precision arithmetic (see [5] and [31, Chapter 5]), are not resolved by the methodology proposed in this work.

The paper is structured as follows. In section 2 the propagation of local rounding errors in standard preconditioned CG, Chronopoulos/Gear CG, and the pipelined CG algorithm is analyzed. Bounds for the gap between the explicitly computed residual and the recursive residual are derived. Section 2.4 proposes an approximate and practically usable estimate for the residual gap. Furthermore, the incorporation of a residual replacement strategy in the pipelined CG method is discussed in section 3. A criterion for automated residual replacement based on the aforementioned error propagation model is suggested. Extensive numerical experiments in section 4 illustrate the error estimate and show the possible improvement in attainable accuracy for the pipelined CG method with automated residual replacement. Parallel scaling results indicate that the residual replacement strategy does not affect the parallel scalability of the pipelined CG method. Finally, conclusions are formulated in section 5.

2. Analysis of local rounding error propagation in variants of the CG algorithm. The analysis in this section is based upon the rounding error analysis performed by Greenbaum in [24] and Strakoš & Gutknecht in [27]. Additional work on this topic can be found in [12, 23, 33, 35, 36, 37, 38, 45, 46]. We assume the following classical model for floating point arithmetic on a machine with machine precision ϵ:

$$\mathrm{fl}(a \pm b) = a(1+\epsilon_1) \pm b(1+\epsilon_2), \quad |\epsilon_1|, |\epsilon_2| \leq \epsilon \tag{2.1}$$

$$\mathrm{fl}(a \text{ op } b) = (a \text{ op } b)(1+\epsilon_3), \quad |\epsilon_3| \leq \epsilon, \quad \text{op} = *, / \tag{2.2}$$

where $\mathrm{fl}(a)$ denotes the finite precision floating point representation of the mathematical quantity a. Under this model, and discarding terms involving ϵ^2 or higher powers of ϵ when terms of order ϵ are present, the following standard results for operations on an n-by-n matrix A, n-length vectors v and w, and scalar number α hold:

$$\|\alpha v - \mathrm{fl}(\alpha v)\| \leq \|\alpha v\|\,\epsilon = |\alpha|\,\|v\|\,\epsilon \tag{2.3}$$

$$\|v + w - \mathrm{fl}(v+w)\| \leq (\|v\| + \|w\|)\,\epsilon \tag{2.4}$$

$$|\,(v,w) - \mathrm{fl}(\,(v,w)\,)| \leq n\,\|v\|\,\|w\|\epsilon \tag{2.5}$$

$$\|Av - \mathrm{fl}(Av)\| \leq (\mu\sqrt{n})\,\|A\|\,\|v\|\,\epsilon \tag{2.6}$$

where μ is the maximum number of nonzeros in any row of A. The norm $\|\cdot\|$ denotes the Euclidean 2-norm throughout this manuscript, unless explicitly stated otherwise.

2.1. Accumulation of local rounding errors in classical CG. Classical preconditioned CG is given by Algorithm 1. Note that in the unpreconditioned case, line 8 is dropped, and each occurrence of u_i is replaced by r_i. In finite precision arithmetic, the recurrences for the computed search direction $\bar{p}_i$, iterate $\bar{x}_i$, and residual $\bar{r}_i$ in iteration i ($i = 0, 1, 2, \ldots$) of the CG algorithm are

$$\begin{aligned} \bar{p}_{i+1} &= \bar{u}_{i+1} + \bar{\beta}_{i+1}\bar{p}_i + \delta_i^p, \\ \bar{x}_{i+1} &= \bar{x}_i + \bar{\alpha}_i\bar{p}_i + \delta_i^x, \\ \bar{r}_{i+1} &= \bar{r}_i - \bar{\alpha}_i A\bar{p}_i + \delta_i^r, \end{aligned} \tag{2.7}$$

Algorithm 1. Preconditioned CG.

```
1: procedure PREC-CG(A, M^{-1}, b, x_0)
2:     r_0 := b − Ax_0; u_0 := M^{-1}r_0 ; p_0 = u_0
3:     for i = 0, ... do
4:         s_i := Ap_i
5:         α_i := (r_i, u_i) / (s_i, p_i)
6:         x_{i+1} := x_i + α_i p_i
7:         r_{i+1} := r_i − α_i s_i
8:         u_{i+1} := M^{-1} r_{i+1}
9:         β_{i+1} := (r_{i+1}, u_{i+1}) / (r_i, u_i)
10:        p_{i+1} := u_{i+1} + β_{i+1} p_i
11:    end for
12: end procedure
```

where δ_i^p, δ_i^r, and δ_i^x contain the local rounding errors produced in step i. In our notation barred variables (e.g., $\bar{p}_i$, $\bar{x}_i$, $\bar{r}_i$, $\bar{u}_i$, $\bar{\alpha}_i$ and $\bar{\beta}_{i+1}$) will always denote the actually computed quantities. This notation should avoid confusion with the mathematical quantities defined in exact arithmetic (e.g., r_i, p_i, and α_i) which are unavailable in practice. Vectors obtained by actually applying the matrix-vector product will be referred to as *explicit* quantities (e.g., $b - A\bar{x}_i$ is called the *explicit* or *true residual*), in contrast to the *recursive* quantities given by (2.7) (e.g., $\bar{r}_i$ is called the *recursive residual*). Since the SPMV is a computationally costly operation, the residual is only computed recursively in Algorithm 1, except for $\bar{r}_0 = \mathrm{fl}(b - A\bar{x}_0)$.

The iteration i local rounding errors satisfy the following bounds:

$$\begin{aligned} \|\delta_i^p\| &\le \left(\|\bar{u}_{i+1}\| + 2\,|\bar{\beta}_{i+1}|\,\|\bar{p}_i\|\right)\epsilon, \\ \|\delta_i^x\| &\le \left(\|\bar{x}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{p}_i\|\right)\epsilon, \\ \|\delta_i^r\| &\le \left(\|\bar{r}_i\| + (\mu\sqrt{n}+2)\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{p}_i\|\right)\epsilon. \end{aligned} \tag{2.8}$$

We now want to estimate the difference (or gap) between the true residual $b - A\bar{x}_i$ and the recursive residual $\bar{r}_i$. Hence, we define this gap as

$$f_i = (b - A\bar{x}_i) - \bar{r}_i. \tag{2.9}$$

The residual $\bar{r}_0$ is computed explicitly in Algorithm 1, and the gap f_0 is thus the round-off from computing $\bar{r}_0$ from A, $\bar{x}_0$ and b, i.e., $f_0 = b - A\bar{x}_0 - \mathrm{fl}(b - A\bar{x}_0)$. The norm of this initial gap is bounded by

$$\|f_0\| \le \left((\mu\sqrt{n}+1)\,\|A\|\,\|\bar{x}_0\| + \|b\|\right)\epsilon. \tag{2.10}$$

In iteration i we obtain the following formula for the gap by substituting the recursions (2.7):

$$\begin{aligned} f_{i+1} &= (b - A\bar{x}_{i+1}) - \bar{r}_{i+1} \\ &= b - A(\bar{x}_i + \bar{\alpha}_i\bar{p}_i + \delta_i^x) - (\bar{r}_i - \bar{\alpha}_i A\bar{p}_i + \delta_i^r) \\ &= f_i - A\delta_i^x - \delta_i^r. \end{aligned} \tag{2.11}$$

This recursive formulation relates the residual error f_{i+1} in step i to the previous residual error f_i. By taking norms on both sides, we obtain an upper bound on $\|f_{i+1}\|$ in function of the previous gap $\|f_i\|$:

$$\|f_{i+1}\| \le \|f_i\| + \|A\delta_i^x + \delta_i^r\|, \tag{2.12}$$

where we can use the bounds (2.8) to further rewrite the right-hand side bound as

$$\begin{aligned}\|A\delta_i^x + \delta_i^r\| &\leq \|A\|\,\|\delta_i^x\| + \|\delta_i^r\| \\ &\leq \left(\|A\|\,\|\bar{x}_i\| + 2\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{p}_i\| + \|\bar{r}_i\| + (\mu\sqrt{n}+2)\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{p}_i\|\right)\epsilon \\ &= \left(\|A\|\,\|\bar{x}_i\| + (\mu\sqrt{n}+4)\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{p}_i\| + \|\bar{r}_i\|\right)\epsilon \\ &:= e_i^f\,\epsilon. \end{aligned} \tag{2.13}$$

Hence, with the above definition of the upper bound factor e_i^f, we obtain

$$\|f_{i+1}\| \leq \|f_i\| + e_i^f\,\epsilon, \tag{2.14}$$

which gives an upper bound on the norm of the gap between the true and recursive residual in any iteration of the method based on the gap in the previous iteration.

The recurrence (2.11) implies that in the classical CG method the gap f_{i+1} is the sum of local rounding errors, i.e.,

$$f_{i+1} = f_0 - \sum_{j=0}^{i} \left(A\delta_j^x + \delta_j^r\right). \tag{2.15}$$

Hence, no amplification of local rounding errors occurs in classical CG since (2.15) indicates that local rounding errors are simply accumulated; see also [24, 27, 41].

2.2. Propagation of local rounding errors in Chronopoulos/Gear CG. In so-called Chronopoulos/Gear CG (commonly denoted CG-CG in this manuscript), Algorithm 2, an extra recurrence for the auxiliary variable s_i is introduced, which in exact arithmetic equals Ap_i, and the auxiliary vectors $w_i = Au_i$ and $u_i = M^{-1}r_i$ are computed explicitly in each iteration, i.e., $\bar{w}_i = \mathrm{fl}(A\bar{u}_i)$ and $\bar{u}_i = \mathrm{fl}(M^{-1}\bar{r}_i)$. Algorithm 2 "avoids" communication by reducing the two global reduction phases of classical CG to one global synchronization (lines 11–12). The unpreconditioned version of the algorithm can be obtained by simply removing line 9 and replacing u_i by r_i where required. In finite precision arithmetic the corresponding recurrences in Algorithm 2 are

$$\begin{aligned} \bar{x}_{i+1} &= \bar{x}_i + \bar{\alpha}_i\bar{p}_i + \delta_i^x, & \bar{p}_i &= \bar{u}_i + \bar{\beta}_i\bar{p}_{i-1} + \delta_i^p, \\ \bar{r}_{i+1} &= \bar{r}_i - \bar{\alpha}_i\bar{s}_i + \delta_i^r, & \bar{s}_i &= A\bar{u}_i + \bar{\beta}_i\bar{s}_{i-1} + \delta_i^s, \end{aligned} \tag{2.16}$$

where the local rounding errors satisfy

$$\begin{aligned} \|\delta_i^x\| &\leq \left(\|\bar{x}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{p}_i\|\right)\epsilon, \\ \|\delta_i^r\| &\leq \left(\|\bar{r}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{s}_i\|\right)\epsilon, \\ \|\delta_i^p\| &\leq \left((\tilde{\mu}\sqrt{n}+1)\,\|M^{-1}\|\,\|\bar{r}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{p}_{i-1}\|\right)\epsilon, \\ \|\delta_i^s\| &\leq \left((\mu\sqrt{n}+\tilde{\mu}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|\bar{r}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{s}_{i-1}\|\right)\epsilon, \end{aligned} \tag{2.17}$$

where $\tilde{\mu}$ is the maximum number of nonzeros in any row of the operator M^{-1}. To estimate the gap between the true and recursive residual we again substitute the recursions (2.16) in $f_i = (b - A\bar{x}_i) - \bar{r}_i$. Note that we have

$$\|f_0\| \leq \left((\mu\sqrt{n}+1)\,\|A\|\,\|\bar{x}_0\| + \|b\|\right)\epsilon \tag{2.18}$$

since the recursion for $\bar{x}_i$ in CG-CG is identical to that in CG; see (2.7). The gap in iteration i is given by

$$\begin{aligned} f_{i+1} &= (b - A\bar{x}_{i+1}) - \bar{r}_{i+1} \\ &= b - A(\bar{x}_i + \bar{\alpha}_i\bar{p}_i + \delta_i^x) - (\bar{r}_i - \bar{\alpha}_i\bar{s}_i + \delta_i^r) \\ &= f_i - \bar{\alpha}_i g_i - A\delta_i^x - \delta_i^r, \end{aligned} \tag{2.19}$$

Algorithm 2. Preconditioned Chronopoulos/Gear CG.

procedure PREC-CG-CG(A, M^{-1}, b, x_0)
  $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $w_0 := Au_0$
  $\alpha_0 := (r_0, u_0)/(w_0, u_0)$; $\beta_0 := 0$; $\gamma_0 := (r_0, u_0)$
  for $i = 0, \ldots$ **do**
    $p_i := u_i + \beta_i p_{i-1}$
    $s_i := w_i + \beta_i s_{i-1}$
    $x_{i+1} := x_i + \alpha_i p_i$
    $r_{i+1} := r_i - \alpha_i s_i$
    $u_{i+1} := M^{-1} r_{i+1}$
    $w_{i+1} := Au_{i+1}$
    $\gamma_{i+1} := (r_{i+1}, u_{i+1})$
    $\delta := (w_{i+1}, u_{i+1})$
    $\beta_{i+1} := \gamma_{i+1}/\gamma_i$
    $\alpha_{i+1} := (\delta/\gamma_{i+1} - \beta_{i+1}/\alpha_i)^{-1}$
  end for
end procedure

where $g_i = A\bar{p}_i - \bar{s}_i$, that is, the gap between the explicit quantity $A\bar{p}_i$ and the recursively computed auxiliary variable $\bar{s}_i$. For the latter gap, it holds for $i = 0$ that

$$\|g_0\| \leq \mu\sqrt{n}\,\|A\|\,\|\bar{p}_0\|\,\epsilon. \tag{2.20}$$

Indeed, in iteration $i = 0$, the gap g_0 is the round-off on the explicit computation $A\bar{p}_0$, i.e., $g_0 = A\bar{p}_0 - \mathrm{fl}(A\bar{p}_0)$. In iteration $i > 0$ the variable $\bar{s}_i$ is computed recursively, and it holds that

$$\begin{aligned} g_i &= A\bar{p}_i - \bar{s}_i \\ &= A(\bar{u}_i + \bar{\beta}_i\bar{p}_{i-1} + \delta_i^p) - (A\bar{u}_i + \bar{\beta}_i\bar{s}_{i-1} + \delta_i^s) \\ &= \bar{\beta}_i g_{i-1} + A\delta_i^p - \delta_i^s. \end{aligned} \tag{2.21}$$

The residual gap is thus given by the coupled recursions

$$\begin{bmatrix} f_{i+1} \\ g_i \end{bmatrix} = \begin{bmatrix} 1 & -\bar{\alpha}_i\bar{\beta}_i \\ 0 & \bar{\beta}_i \end{bmatrix} \begin{bmatrix} f_i \\ g_{i-1} \end{bmatrix} + \begin{bmatrix} -A\delta_i^x - \delta_i^r - \bar{\alpha}_i\,(A\delta_i^p - \delta_i^s) \\ A\delta_i^p - \delta_i^s \end{bmatrix}. \tag{2.22}$$

Taking norms, we obtain the upper bounds

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \end{bmatrix} \leq \begin{bmatrix} 1 & |\bar{\alpha}_i\bar{\beta}_i| \\ 0 & |\bar{\beta}_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \end{bmatrix} + \begin{bmatrix} \|A\delta_i^x + \delta_i^r\| + |\bar{\alpha}_i|\|A\delta_i^p - \delta_i^s\| \\ \|A\delta_i^p - \delta_i^s\| \end{bmatrix}. \tag{2.23}$$

This bound can be further rewritten into more tractable expressions using the bounds for the local rounding errors in (2.17), i.e.,

$$\begin{aligned} \|A\delta_i^x + \delta_i^r\| &\leq \|A\|\,\|\delta_i^x\| + \|\delta_i^r\| \\ &\leq (\|A\|\,\|\bar{x}_i\| + 2\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{p}_i\| + \|\bar{r}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{s}_i\|)\,\epsilon \\ &:= e_i^f\,\epsilon \end{aligned} \tag{2.24}$$

and

$$\begin{aligned}\|A\delta_i^p - \delta_i^s\| &\le \|A\|\,\|\delta_i^p\| + \|\delta_i^s\| \\ &\le \left((\tilde{\mu}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|\bar{r}_i\| + 2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{p}_{i-1}\|\right. \\ &\quad \left. + (\mu\sqrt{n}+\tilde{\mu}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|\bar{r}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{s}_{i-1}\|\right)\epsilon \\ &= \left(2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{p}_{i-1}\| + ((\mu+2\tilde{\mu})\sqrt{n}+2)\,\|A\|\,\|M^{-1}\|\,\|\bar{r}_i\|\right. \\ &\quad \left. + 2\,|\bar{\beta}_i|\,\|\bar{s}_{i-1}\|\right)\epsilon \\ &:= e_i^g\epsilon. \end{aligned} \tag{2.25}$$

Note that the definitions of the bounds are local to each subsection; definition (2.24) above holds for CG-CG and should not be confused with the earlier identical notation defined by (2.13) for the bound in classical CG. Hence, with the factors e_i^f and e_i^g defined as above, the norm of the gap between the true and recursive residual is bounded by the recursively defined system of upper bounds

$$\begin{bmatrix}\|f_{i+1}\| \\ \|g_i\|\end{bmatrix} \le \begin{bmatrix}1 & |\bar{\alpha}_i\bar{\beta}_i| \\ 0 & |\bar{\beta}_i|\end{bmatrix}\begin{bmatrix}\|f_i\| \\ \|g_{i-1}\|\end{bmatrix} + \begin{bmatrix}e_i^f\epsilon + |\bar{\alpha}_i|\,e_i^g\epsilon \\ e_i^g\epsilon\end{bmatrix}. \tag{2.26}$$

From (2.22) it can be derived by induction that the residual gap in iteration i is

$$f_{i+1} = f_0 - \sum_{j=0}^{i}\left(A\delta_j^x + \delta_j^r\right) - \sum_{j=0}^{i}\bar{\alpha}_j\left[\left(\prod_{k=1}^{j}\bar{\beta}_k\right)g_0 + \sum_{k=1}^{j}\left(\prod_{l=k+1}^{j}\bar{\beta}_l\right)(A\delta_k^p - \delta_k^s)\right]. \tag{2.27}$$

Note that this is in sharp contrast to the error behavior of the residual gap in the classical CG algorithm, where the gap after $i+1$ steps is a simple sum of local rounding errors; see (2.15). Indeed, the local rounding errors $(A\delta_k^p - \delta_k^s)$ $(1 \le k \le j)$ that contribute to the difference $f_{i+1} = (b - A\bar{x}_{i+1}) - \bar{r}_{i+1}$ in (2.27) are potentially amplified by the factors $\prod_{l=k+1}^{j}\bar{\beta}_l$. Note that in exact arithmetic this product is

$$\prod_{l=k+1}^{j}\beta_l = \frac{\|r_j\|^2}{\|r_k\|^2}, \quad 1 \le k \le j, \tag{2.28}$$

which may be large for some $k \le j$. Consequently, like the three-term recurrence CG algorithm [43] which was analyzed in [27] (see also [5]), the CG-CG method may suffer from a dramatic amplification of local rounding errors throughout the algorithm, and the accuracy achieved by Algorithm 2 can be significantly worse compared to Algorithm 1.

2.3. Propagation of local rounding errors in pipelined CG. In preconditioned pipelined CG, Algorithm 3, additional recurrences are introduced for the auxiliary variables w_i, z_i, u_i, and q_i, which, respectively, equal Au_i, Aq_i, $M^{-1}r_i$, and $M^{-1}s_i$ in exact arithmetic, whereas $v_i = Am_i$ and $m_i = M^{-1}w_i$ are computed explicitly. In addition to reducing communication, Algorithm 3 "hides" the communication phase (lines 4–5) behind the SPMV and preconditioner application (lines 6–7). Replacing the recurrences by their finite precision equivalents, we have

$$\begin{aligned} \bar{x}_{i+1} &= \bar{x}_i + \bar{\alpha}_i\bar{p}_i + \delta_i^x, & \bar{p}_i &= \bar{u}_i + \bar{\beta}_i\bar{p}_{i-1} + \delta_i^p, \\ \bar{r}_{i+1} &= \bar{r}_i - \bar{\alpha}_i\bar{s}_i + \delta_i^r, & \bar{s}_i &= \bar{w}_i + \bar{\beta}_i\bar{s}_{i-1} + \delta_i^s, \\ \bar{w}_{i+1} &= \bar{w}_i - \bar{\alpha}_i\bar{z}_i + \delta_i^w, & \bar{z}_i &= A\bar{m}_i + \bar{\beta}_i\bar{z}_{i-1} + \delta_i^z, \\ \bar{u}_{i+1} &= \bar{u}_i - \bar{\alpha}_i\bar{q}_i + \delta_i^u, & \bar{q}_i &= \bar{m}_i + \bar{\beta}_i\bar{q}_{i-1} + \delta_i^q, \end{aligned} \tag{2.29}$$

Algorithm 3. Preconditioned pipelined CG.

procedure PREC-P-CG(A, M^{-1}, b, x_0)
  $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $w_0 := Au_0$
  for $i = 0, \ldots$ **do**
    $\gamma_i := (r_i, u_i)$
    $\delta := (w_i, u_i)$
    $m_i := M^{-1}w_i$
    $v_i := Am_i$
    if $i > 0$ **then**
      $\beta_i := \gamma_i/\gamma_{i-1}$; $\alpha_i := (\delta/\gamma_i - \beta_i/\alpha_{i-1})^{-1}$
    else
      $\beta_i := 0$; $\alpha_i := \gamma_i/\delta$
    end if
    $z_i := v_i + \beta_i z_{i-1}$
    $q_i := m_i + \beta_i q_{i-1}$
    $s_i := w_i + \beta_i s_{i-1}$
    $p_i := u_i + \beta_i p_{i-1}$
    $x_{i+1} := x_i + \alpha_i p_i$
    $r_{i+1} := r_i - \alpha_i s_i$
    $u_{i+1} := u_i - \alpha_i q_i$
    $w_{i+1} := w_i - \alpha_i z_i$
  end for
end procedure

where the local rounding errors are bounded by

$$
\begin{aligned}
\|\delta_i^x\| &\le (\|\bar{x}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{p}_i\|)\,\epsilon,\\
\|\delta_i^p\| &\le \left(\|\bar{u}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{p}_{i-1}\|\right)\epsilon,\\
\|\delta_i^r\| &\le (\|\bar{r}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{s}_i\|)\,\epsilon,\\
\|\delta_i^s\| &\le \left(\|\bar{w}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{s}_{i-1}\|\right)\epsilon,\\
\|\delta_i^w\| &\le (\|\bar{w}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{z}_i\|)\,\epsilon,\\
\|\delta_i^z\| &\le \left((\mu\sqrt{n} + \tilde{\mu}\sqrt{n} + 1)\,\|A\|\,\|M^{-1}\|\,\|\bar{w}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{z}_{i-1}\|\right)\epsilon,\\
\|\delta_i^u\| &\le (\|\bar{u}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{q}_i\|)\,\epsilon,\\
\|\delta_i^q\| &\le \left((\tilde{\mu}\sqrt{n} + 1)\,\|M^{-1}\|\,\|\bar{w}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{q}_{i-1}\|\right)\epsilon.
\end{aligned}
\tag{2.30}
$$

The gap $f_i = (b - A\bar{x}_i) - \bar{r}_i$ can then be calculated similarly to (2.19). The initial gap f_0 satisfies (2.18), and in iteration i we have

$$
\begin{aligned}
f_{i+1} &= (b - A\bar{x}_{i+1}) - \bar{r}_{i+1}\\
&= b - A(\bar{x}_i + \bar{\alpha}_i\bar{p}_i + \delta_i^x) - (\bar{r}_i - \bar{\alpha}_i\bar{s}_i + \delta_i^r)\\
&= f_i - \bar{\alpha}_i g_i - A\delta_i^x - \delta_i^r.
\end{aligned}
\tag{2.31}
$$

The residual gap is again coupled to $g_i = A\bar{p}_i - \bar{s}_i$, which can be written as

$$
\begin{aligned}
g_i &= A\bar{p}_i - \bar{s}_i\\
&= A(\bar{u}_i + \bar{\beta}_i\bar{p}_{i-1} + \delta_i^p) - (\bar{w}_i + \bar{\beta}_i\bar{s}_{i-1} + \delta_i^s)\\
&= h_i + \bar{\beta}_i g_{i-1} + A\delta_i^p - \delta_i^s,
\end{aligned}
\tag{2.32}
$$

where g_0 satisfies (2.20) and we define $h_i = A\bar{u}_i - \bar{w}_i$. Instead of being computed explicitly, the auxiliary variable $\bar{w}_i$ is also computed recursively in pipelined CG, leading to an additional coupling of the residual gap f_i to the difference h_i. For $i = 0$, it holds that the norm of the gap h_i is bounded by

$$\|h_0\| \leq \mu\sqrt{n}\,\|A\|\,\|\bar{u}_0\|\,\epsilon. \tag{2.33}$$

Substituting the recurrences (2.29), we find that the gap between $A\bar{u}_{i+1}$ and $\bar{w}_{i+1}$ in iteration i is

$$\begin{aligned} h_{i+1} &= A\bar{u}_{i+1} - \bar{w}_{i+1} \\ &= A(\bar{u}_i - \bar{\alpha}_i\bar{q}_i + \delta_i^u) - (\bar{w}_i - \bar{\alpha}_i\bar{z}_i + \delta_i^w) \\ &= h_i - \bar{\alpha}_i j_i + A\delta_i^u - \delta_i^w, \end{aligned} \tag{2.34}$$

which relates the residual error to the error $j_i = A\bar{q}_i - \bar{z}_i$. The latter gap is due to the recursive computation of the auxiliary variable $\bar{z}_i$. For $i = 0$, we can bound the norm of j_i by the norm of the round-off, i.e.,

$$\|j_0\| \leq \mu\sqrt{n}\,\|A\|\,\|\bar{q}_0\|\,\epsilon. \tag{2.35}$$

Using again the recursive definitions (2.29), we obtain

$$\begin{aligned} j_i &= A\bar{q}_i - \bar{z}_i \\ &= A(\bar{m}_i + \bar{\beta}_i\bar{q}_{i-1} + \delta_i^q) - (A\bar{m}_i + \bar{\beta}_i\bar{z}_{i-1} + \delta_i^z) \\ &= \bar{\beta}_i j_{i-1} + A\delta_i^q - \delta_i^z. \end{aligned} \tag{2.36}$$

Hence, for pipelined CG, the residual gap is given by the system of coupled equations

$$\begin{bmatrix} f_{i+1} \\ g_i \\ h_{i+1} \\ j_i \end{bmatrix} = \begin{bmatrix} 1 & -\bar{\alpha}_i\bar{\beta}_i & -\bar{\alpha}_i & 0 \\ 0 & \bar{\beta}_i & 1 & 0 \\ 0 & 0 & 1 & -\bar{\alpha}_i\bar{\beta}_i \\ 0 & 0 & 0 & \bar{\beta}_i \end{bmatrix} \begin{bmatrix} f_i \\ g_{i-1} \\ h_i \\ j_{i-1} \end{bmatrix} + \begin{bmatrix} -A\delta_i^x - \delta_i^r - \bar{\alpha}_i\,(A\delta_i^p - \delta_i^s) \\ A\delta_i^p - \delta_i^s \\ A\delta_i^u - \delta_i^w - \bar{\alpha}_i\,(A\delta_i^q - \delta_i^z) \\ A\delta_i^q - \delta_i^z \end{bmatrix}. \tag{2.37}$$

Taking norms of both sides in (2.37), we arrive at the following coupled system of upper bounds for the gaps in pipelined CG:

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \\ \|h_{i+1}\| \\ \|j_i\| \end{bmatrix} \leq \begin{bmatrix} 1 & |\bar{\alpha}_i\bar{\beta}_i| & |\bar{\alpha}_i| & 0 \\ 0 & |\bar{\beta}_i| & 1 & 0 \\ 0 & 0 & 1 & |\bar{\alpha}_i\bar{\beta}_i| \\ 0 & 0 & 0 & |\bar{\beta}_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \\ \|h_i\| \\ \|j_{i-1}\| \end{bmatrix} + \begin{bmatrix} \|A\delta_i^x + \delta_i^r\| + |\bar{\alpha}_i|\|A\delta_i^p - \delta_i^s\| \\ \|A\delta_i^p - \delta_i^s\| \\ \|A\delta_i^u - \delta_i^w\| + |\bar{\alpha}_i|\|A\delta_i^q - \delta_i^z\| \\ \|A\delta_i^q - \delta_i^z\| \end{bmatrix}. \tag{2.38}$$

The per-iteration additions on the right-hand side in (2.38) can be bounded further using the local error bounds (2.30). We hence obtain the computable bounds

$$\begin{aligned} \|A\delta_i^x + \delta_i^r\| &\leq \|A\|\,\|\delta_i^x\| + \|\delta_i^r\| \\ &\leq (\|A\|\,\|\bar{x}_i\| + 2\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{p}_i\| + \|\bar{r}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{s}_i\|)\,\epsilon \\ &:= e_i^f\,\epsilon \end{aligned} \tag{2.39}$$

$$\begin{aligned} \|A\delta_i^p - \delta_i^s\| &\leq \|A\|\,\|\delta_i^p\| + \|\delta_i^s\| \\ &\leq \left(\|A\|\,\|\bar{u}_i\| + 2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{p}_{i-1}\| + \|\bar{w}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{s}_{i-1}\|\right)\epsilon \\ &:= e_i^g\,\epsilon \end{aligned} \tag{2.40}$$

$$\begin{aligned}\|A\delta_i^u - \delta_i^w\| &\leq \|A\|\,\|\delta_i^u\| + \|\delta_i^w\| \\ &\leq \left(\|A\|\,\|\bar{u}_i\| + 2\,|\bar{\alpha}_i|\,\|A\|\,\|\bar{q}_i\| + \|\bar{w}_i\| + 2\,|\bar{\alpha}_i|\,\|\bar{z}_i\|\right)\epsilon \\ &:= e_i^h\epsilon \end{aligned} \tag{2.41}$$

$$\begin{aligned}\|A\delta_i^q - \delta_i^z\| &\leq \|A\|\,\|\delta_i^q\| + \|\delta_i^z\| \\ &\leq \left((\tilde{\mu}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|\bar{w}_i\| + 2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{q}_{i-1}\|\right. \\ &\quad \left. + (\mu\sqrt{n} + \tilde{\mu}\sqrt{n}+1)\,\|A\|\,\|M^{-1}\|\,\|\bar{w}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{z}_{i-1}\|\right)\epsilon \\ &= \left(2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{q}_{i-1}\| + ((\mu+2\tilde{\mu})\sqrt{n}+2)\,\|A\|\,\|M^{-1}\|\,\|\bar{w}_i\|\right. \\ &\quad \left. + 2\,|\bar{\beta}_i|\,\|\bar{z}_{i-1}\|\right)\epsilon \\ &:= e_i^j\epsilon. \end{aligned} \tag{2.42}$$

This yields the following system of upper bounds on the norm of the gap between the true and recursive residual:

$$\begin{bmatrix}\|f_{i+1}\| \\ \|g_i\| \\ \|h_{i+1}\| \\ \|j_i\|\end{bmatrix} \leq \begin{bmatrix}1 & |\bar{\alpha}_i\bar{\beta}_i| & |\bar{\alpha}_i| & 0 \\ 0 & |\bar{\beta}_i| & 1 & 0 \\ 0 & 0 & 1 & |\bar{\alpha}_i\bar{\beta}_i| \\ 0 & 0 & 0 & |\bar{\beta}i|\end{bmatrix}\begin{bmatrix}\|f_i\| \\ \|g_{i-1}\| \\ \|h_i\| \\ \|j_{i-1}\|\end{bmatrix} + \begin{bmatrix}e_i^f\epsilon + |\bar{\alpha}_i|e_i^g\epsilon \\ e_i^g\epsilon \\ e_i^h\epsilon + |\bar{\alpha}_i|e_i^j\epsilon \\ e_i^j\epsilon\end{bmatrix}, \tag{2.43}$$

where e_i^f, e_i^g, e_i^h, and e_i^j are defined in (2.39)–(2.42).

By induction, (2.37) can be reformulated into an expression for the residual gap f_{i+1} with respect to f_0, g_0, h_0, j_0, and local rounding errors, similar to (2.27), i.e.,

$$f_{i+1} = f_0 - \sum_{j=0}^{i}\bar{\alpha}_j g_j - \sum_{j=0}^{i}\left(A\delta_j^x + \delta_j^r\right), \tag{2.44}$$

where

$$g_j = \left(\prod_{k=1}^{j}\bar{\beta}_k\right)g_0 + \sum_{k=1}^{j}\left(\prod_{l=k+1}^{j}\bar{\beta}_l\right)(A\delta_k^p - \delta_k^s) + \sum_{k=1}^{j}\left(\prod_{l=k+1}^{j}\bar{\beta}_l\right)h_k, \tag{2.45}$$

with

$$h_k = h_0 - \sum_{l=0}^{k-1}\bar{\alpha}_l j_l + \sum_{l=0}^{k-1}\left(A\delta_l^u - \delta_l^w\right), \tag{2.46}$$

and where

$$j_l = \left(\prod_{m=1}^{l}\bar{\beta}_m\right)j_0 + \sum_{m=1}^{l}\left(\prod_{n=m+1}^{l}\bar{\beta}_n\right)(A\delta_m^q - \delta_m^z). \tag{2.47}$$

It is clear from (2.44)–(2.47) that due to the extra recurrence relations, the propagation of local rounding errors may be even more dramatic for p-CG since $(A\delta_k^p - \delta_k^s)$ $(1 \leq k \leq j)$, $(A\delta_l^u - \delta_l^w)$ $(0 \leq l \leq k-1)$, and $(A\delta_m^q - \delta_m^z)$ $(1 \leq m \leq l)$ are all potentially amplified during the algorithm. This may lead to significantly reduced maximal attainable accuracy compared to both classical CG and CG-CG.

Note that the auxiliary variables u_i and q_i, which in exact arithmetic represent the preconditioned versions of the residual r_i and the auxiliary variable s_i, respectively,

are also computed recursively in pipelined CG. Hence, we can write down an analogous derivation for the gap between the explicit and recursive preconditioned residual, that is, $k_i = M^{-1}\bar{r}_i - \bar{u}_i$. For $i = 0$ we have

$$\|k_0\| \leq \tilde{\mu}\sqrt{n}\,\|M^{-1}\|\,\|\bar{r}_0\|\,\epsilon, \tag{2.48}$$

and in iteration i we find

$$\begin{aligned} k_{i+1} &= M^{-1}\bar{r}_{i+1} - \bar{u}_{i+1} \\ &= M^{-1}(\bar{r}_i - \bar{\alpha}_i\bar{s}_i + \delta_i^r) - (\bar{u}_i - \bar{\alpha}_i\bar{q}_i + \delta_i^u) \\ &= k_i - \bar{\alpha}_i\ell_i + M^{-1}\delta_i^r - \delta_i^u, \end{aligned} \tag{2.49}$$

where we define $\ell_i = M^{-1}\bar{s}_i - \bar{q}_i$. Finally, for the gap between the explicit quantity $M^{-1}\bar{s}_i$ and the recursive variable $\bar{q}_i$, we have for $i = 0$ that

$$\|\ell_0\| \leq \tilde{\mu}\sqrt{n}\,\|M^{-1}\|\,\|\bar{s}_0\|\,\epsilon. \tag{2.50}$$

By once again inserting the recurrences from (2.29), we find that ℓ_i in iteration i is

$$\begin{aligned} \ell_i &= M^{-1}\bar{s}_i - \bar{q}_i \\ &= M^{-1}(\bar{w}_i + \bar{\beta}_i\bar{s}_{i-1} + \delta_i^s) - (\bar{m}_i + \bar{\beta}_i\bar{q}_{i-1} + \delta_i^q) \\ &= \bar{\beta}_i\ell_{i-1} + M^{-1}\delta_i^s - \delta_i^q. \end{aligned} \tag{2.51}$$

The last equation in (2.51) holds since $\bar{m}_i$ is computed explicitly as $M^{-1}\bar{w}_i$ in Algorithm 3. This leads to a separate system of coupled recurrences for the gap on the preconditioned residual k_{i+1}:

$$\begin{bmatrix} k_{i+1} \\ \ell_i \end{bmatrix} = \begin{bmatrix} 1 & -\bar{\alpha}_i\bar{\beta}_i \\ 0 & \bar{\beta}_i \end{bmatrix} \begin{bmatrix} k_i \\ \ell_{i-1} \end{bmatrix} + \begin{bmatrix} M^{-1}\delta_i^r - \delta_i^u - \bar{\alpha}_i\left(M^{-1}\delta_i^s - \delta_i^q\right) \\ M^{-1}\delta_i^s - \delta_i^q \end{bmatrix}. \tag{2.52}$$

However, since the gap k_{i+1} is uncoupled from the residual gap f_{i+1}, the coupled recurrences (2.49)–(2.51) are not be taken into account when establishing bounds for the norm of the residual gap in (2.43).

2.4. A practical estimate for the residual gap. In the previous sections we have derived upper bounds for the norm of the residual gap for several variants of the CG algorithm. Although insightful from an analytical perspective, these bounds are typically not sharp. Indeed, the bounds on the norms of the local rounding errors (2.30) may largely overestimate the actual error norms (see the discussion in [24, p. 541]). For example, the right-hand side of (2.43) could be much larger than the left-hand side and hence could provide a poor estimate for the actual residual gap.

To obtain a more realistic estimate for the residual gap, we turn to statistical analysis of the rounding errors; see [50]. A well-known rule of thumb [30] is that a realistic error estimate can be obtained by replacing the dimension-dependent constants in a rounding error bound by their square root; thus, if the bound is $f(n)\epsilon$, the error is approximately of order $\sqrt{f(n)}\epsilon$. Hence, instead of using the upper bounds (2.39)–(2.42), we use the following approximations for the local error norms:

$$\begin{aligned} \|A\delta_i^x + \delta_i^r\| &\approx \sqrt{e_i^f}\,\epsilon, \qquad & \|A\delta_i^p - \delta_i^s\| &\approx \sqrt{e_i^g}\,\epsilon, \\ \|A\delta_i^u + \delta_i^w\| &\approx \sqrt{e_i^h}\,\epsilon, \qquad & \|A\delta_i^q - \delta_i^z\| &\approx \sqrt{e_i^j}\,\epsilon. \end{aligned} \tag{2.53}$$

Note that for the norms of the initial gaps in (2.18), (2.20), (2.33), and (2.35), a similar square root rescaling of the respective dimension-dependent factors has to be applied. We hence assume that the norm of the residual gap in iteration i of the pipelined CG algorithm can be estimated as follows:

$$\begin{bmatrix} \|f_{i+1}\| \\ \|g_i\| \\ \|h_{i+1}\| \\ \|j_i\| \end{bmatrix} \approx \begin{bmatrix} 1 & |\bar{\alpha}_i\bar{\beta}_i| & |\bar{\alpha}_i| & 0 \\ 0 & |\bar{\beta}_i| & 1 & 0 \\ 0 & 0 & 1 & |\bar{\alpha}_i\bar{\beta}_i| \\ 0 & 0 & 0 & |\bar{\beta}_i| \end{bmatrix} \begin{bmatrix} \|f_i\| \\ \|g_{i-1}\| \\ \|h_i\| \\ \|j_{i-1}\| \end{bmatrix} + \begin{bmatrix} \sqrt{e_i^f}\epsilon + |\bar{\alpha}_i|\sqrt{e_i^g}\epsilon \\ \sqrt{e_i^g}\epsilon \\ \sqrt{e_i^h}\epsilon + |\bar{\alpha}_i|\sqrt{e_i^j}\epsilon \\ \sqrt{e_i^j}\epsilon \end{bmatrix}. \tag{2.54}$$

This approximation tends to yield a good (a posteriori) estimate for the actual residual gap, as illustrated by the numerical experiments in the next section.

A second practical remark concerns the computation of the matrix and preconditioner norms, $\|A\|$ and $\|M^{-1}\|$, in the estimate for the gap between the true and recursive residual. The use of the matrix 2-norm is often prohibited in practice since it is computationally expensive for large-scale systems. However, using the norm inequality $\|A\|_2 \leq \sqrt{n}\,\|A\|_\infty$, the matrix 2-norms in the estimate can be replaced by their respective maximum norms multiplied by $\sqrt{n}$. This slightly worsens the estimate but provides practically computable quantities for e_i^f, e_i^g, e_i^h, and e_i^j. Alternatively, in the context of matrix-free computations, randomized probabilistic techniques for matrix norm computation may be used; see, for example, [28].

A related issue concerns the computation of the norm of the preconditioner. The operator M^{-1} is often not available in matrix form. This is the case when preconditioning the system with, e.g., an Incomplete Cholesky factorization (ICC) type of scheme or any (stencil-based) scheme where M^{-1} is not explicitly formed. For these commonly used preconditioning methods the norm $\|M^{-1}\|$ is unavailable. Explicit use of the preconditioner norm can be avoided by reformulating the local rounding error bounds (2.30) with respect to the preconditioned variable $\bar{m}_i = M^{-1}\bar{w}_i$, that is,

$$\begin{aligned} \|\delta_i^z\| &\leq \left((\mu\sqrt{n}+1)\,\|A\|\,\|\bar{m}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{z}_{i-1}\|\right)\epsilon, \\ \|\delta_i^q\| &\leq \left(\|\bar{m}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{q}_{i-1}\|\right)\epsilon. \end{aligned} \tag{2.55}$$

These bounds do not explicitly take the rounding error of the multiplication $M^{-1}\bar{w}_i$ into account but rather implicitly bound the local rounding error using $\|\bar{m}_i\|$. With these local rounding error bounds, e_i^j can now be defined analogous to (2.42) as

$$\begin{aligned} \|A\delta_i^q - \delta_i^z\| &\leq \|A\|\,\|\delta_i^q\| + \|\delta_i^z\| \\ &\leq \big(\|A\|\,\|\bar{m}_i\| + 2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{q}_{i-1}\| \\ &\quad + (\mu\sqrt{n}+1)\,\|A\|\,\|\bar{m}_i\| + 2\,|\bar{\beta}_i|\,\|\bar{z}_{i-1}\|\big)\,\epsilon \\ &= \left((\mu\sqrt{n}+2)\,\|A\|\,\|\bar{m}_i\| + 2\,|\bar{\beta}_i|\,\|A\|\,\|\bar{q}_{i-1}\| + 2\,|\bar{\beta}_i|\,\|\bar{z}_{i-1}\|\right)\epsilon \\ &:= e_i^j\epsilon. \end{aligned} \tag{2.56}$$

With e_i^j defined as in (2.56), (2.54) can also be used to estimate the residual gap in pipelined CG when the preconditioning matrix M^{-1} is not formed explicitly.

We point out that, as an alternative to computing a residual gap estimate, one could explicitly compute the residual $b - A\bar{x}_i$ in each iteration of the algorithm to keep track of the residual gap. However, calculating $b - A\bar{x}_i$ in each iteration is computationally much too expensive in practice. The computation of the estimate requires only the computation of e_i^f, e_i^g, e_i^h, and e_i^j in each iteration of the algorithm; see (2.54). Note that the scalar expressions (2.39)–(2.42) and (2.56) contain several

norms that are by default not computed in Algorithm 3. However, these norm computations cause no additional communication overhead since they can be combined with the existing global communication phase on lines 4–5. Hence, the computational cost of calculating the residual gap estimate is negligible compared to computing the residual explicitly, and, in contrast to the true residual, the estimate (2.54) can be computed in real time in each iteration of the algorithm without additional computational overhead.

Finally, we note that the norms of some auxiliary variables, notably $\bar{p}_i$, $\bar{s}_i$, $\bar{q}_i$, $\bar{z}_i$, and $\bar{m}_i$, are unavailable in step i since these vectors are not defined until *after* the global reduction phase. Hence, their norms can be computed at the earliest in the global reduction phase of iteration $i+1$. Consequently, in practical implementations the estimated norm $\|f_{i+1}\|$, defined by (2.54), can only be computed in iteration $i+1$. This means that when including the estimates for the residual gap into the pipelined CG algorithm, a delay of one iteration on the estimates is unavoidable.

3. Pipelined CG with automated residual replacement. In this section we propose an automated residual replacement strategy for pipelined CG, based on the estimate for the gap between the true and recursive residual proposed in section 2.4. Although the derivation of our replacement strategy is partially based on heuristics and would certainly benefit from further theoretical investigation, this ad hoc countermeasure aims to improve the possibly dramatically reduced maximal attainable accuracy of the pipelined method. The idea of performing manual residual replacements to increase the attainable accuracy of pipelined CG was already suggested in the original paper [22]. However, to establish an automated replacement strategy, a practically computable criterion for replacement should be available. We suggest such a criterion under the assumption that the orthogonality of the Krylov basis vectors is not (critically) affected by rounding errors. Although this condition may not always hold in practice, it enables designing a heuristic with low computational overhead that proves effective in many situations as described in section 4.

We follow the basic idea of residual replacement in Krylov subspace methods as discussed by Van der Vorst et al. [49] and Sleijpen et al. [40, 42]. In specific iterations of the algorithm, the vectors $\bar{r}_{i+1}$, $\bar{w}_{i+1}$, $\bar{u}_{i+1}$, $\bar{s}_i$, $\bar{z}_i$, and $\bar{q}_i$, which are computed recursively in iteration i, are instead computed explicitly, such that

$$\begin{aligned} \bar{r}_{i+1} &= \mathrm{fl}(b - A\bar{x}_{i+1}), & \bar{u}_{i+1} &= \mathrm{fl}(M^{-1}\bar{r}_{i+1}), & \bar{w}_{i+1} &= \mathrm{fl}(A\bar{u}_{i+1}), \\ \bar{s}_i &= \mathrm{fl}(A\bar{p}_i), & \bar{q}_i &= \mathrm{fl}(M^{-1}\bar{s}_i) & \bar{z}_i &= \mathrm{fl}(A\bar{q}_i). \end{aligned} \tag{3.1}$$

Note how the current iterate $\bar{x}_{i+1}$ and the search direction $\bar{p}_i$ are evidently not replaced since no explicit formulae for these vectors are available.

Two important caveats arise when incorporating a residual replacement in an iterative method. First, one could inquire if such a drastic replacement strategy does not destroy (or delay) convergence. A second, related question concerns the use of a criterion for the iteration in which replacements should take place. Since each residual replacement step comes at an additional cost of computing the SPMVs in (3.1), an accurate criterion to determine the need for residual replacement that does not overestimate the total number of replacements is essential.

We briefly recapitulate the main results from [47] and [49] below. The recurrences for $\bar{r}_{i+1}$ and $\bar{p}_{i+1}$ in the (unpreconditioned) Algorithm 1 are

$$\begin{aligned} \bar{r}_{i+1} &= \bar{r}_i - \bar{\alpha}_i A\bar{p}_i + \delta_i^r, \\ \bar{p}_{i+1} &= \bar{r}_{i+1} + \bar{\beta}_{i+1}\bar{p}_i + \delta_i^p, \end{aligned} \tag{3.2}$$

where δ_i^r and δ_i^p are bounded by (2.8). Combining the above recursions yields the perturbed Lanczos relation

$$AZ_i = Z_i T_i - \frac{\|\bar{r}_0 - \bar{\alpha}_0 A\bar{p}_0\|}{\bar{\alpha}_i \|\bar{r}_1\| \|\bar{r}_i\|} \bar{r}_{i+1} e_i^T + F_i \quad \text{with} \quad Z_i = \left[\frac{\bar{r}_1}{\|\bar{r}_1\|}, \ldots, \frac{\bar{r}_i}{\|\bar{r}_i\|} \right] \tag{3.3}$$

(see [49]), where T_i is a tridiagonal matrix and F_i is a perturbation caused by the local rounding errors. We assume that Z_i is full rank (which might not be true in practice; see below). A key result from [47] states that if $\bar{r}_i$ satisfies the relation (3.3), then

$$\|\bar{r}_{i+1}\| \leq C_i \min_{p \in \mathcal{P}_i, p(0)=1} \|p(A - F_i Z_i^+)\, \bar{r}_1\|, \tag{3.4}$$

where $C_i > 0$ is an iteration-dependent constant. This result suggests that even if the perturbation F_i is significantly larger than ϵ, which is the case after residual replacement, the norm of the residual may not be significantly affected, as illustrated by the experiments presented in [47]. Based on the bound (3.4), Van der Vorst et al. propose in [49] to explicitly update the residuals and other vectors only when the residual norm is sufficiently large compared to the norm of the residual gap. Performing replacements when $\|\bar{r}_i\|$ is small is not recommended since this could negatively affect convergence. Note that although this exposition provides a useful intuition on when to perform residual replacements, it can in general not be considered as a full theoretical validation of the latter since the assumption that Z_i is full rank is often not satisfied in practical computations.

A replacement in step i eliminates the residual gap f_{i+1}. However, it should be carried out before $\|f_i\|$ becomes too large relative to $\|\bar{r}_i\|$. In analogy to [49], we use a threshold τ, typically chosen as $\tau = \sqrt{\epsilon}$, and perform a residual replacement in step i of Algorithm 3 if

$$\|f_{i-1}\| \leq \tau \|\bar{r}_{i-1}\| \quad \text{and} \quad \|f_i\| > \tau \|\bar{r}_i\|. \tag{3.5}$$

This criterion ensures that replacements are only allowed when $\|\bar{r}_i\|$ is sufficiently large with respect to $\|f_i\|$. Furthermore, no excess replacement steps are performed, thus keeping the total computational cost as low as possible. The residual gap model (2.54) allows for the practical implementation of the replacement criterion (3.5) in pipelined CG, see Algorithm 4. Note that criterion (3.5) does not compare the estimate $\|f_{i+1}\|$ to the current residual norm $\|\bar{r}_{i+1}\|$ computed in iteration i since the norms of both these quantities are not yet available; see the discussion near the end of section 2.4. This implies that some additional storage is required for these auxiliary variables between subsequent iterations. The resulting algorithm is called pipelined CG with automated residual replacement (p-CG-rr) and is detailed in Algorithm 4.

In practical computations conditioning of the Lanczos vectors matrix Z_i in (3.3) may be poor due to numerical loss of orthogonality, which may cause the pseudoinverse Z_i^+ in (3.4) to become very ill-conditioned. This somewhat restricts the practical validity of the above argument. We again stress that the residual replacement strategy proposed in this section should be interpreted primarily as a practically usable heuristic that allows to improve accuracy rather than a general and theoretically justified countermeasure to the loss of attainable accuracy. Moreover, we point out that in finite precision arithmetic the residual replacement strategy itself may be a source of rounding errors in the algorithm [5], possibly leading to delayed convergence, see [45]. An in-depth theoretical analysis of the effect of replacements on convergence is, however, beyond the scope of this work.

Algorithm 4. Preconditioned pipelined CG with automated residual replacement.

procedure PREC-P-CG-RR(A, M^{-1}, b, x_0)
  $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $w_0 := Au_0$; $\zeta := \|b\|_2$; $\tau := \sqrt{\epsilon}$
  $n = \text{length}(b)$; $\theta = \sqrt{n}\|A\|_\infty$; $\mu = \max(\text{rowsums}(A))$; replace := false
  for $i = 0, \ldots$ **do**
    $\gamma_i := (r_i, u_i)$; $\delta := (w_i, u_i)$; $\rho_{i+1} := \|r_i\|_2$
    if $i > 0$ **then**
      $\chi_i := \|x_{i-1}\|_2$; $\pi_i := \|p_{i-1}\|_2$; $\sigma_i := \|s_{i-1}\|_2$; $\xi_i := \|u_{i-1}\|_2$
      $\omega_i := \|w_{i-1}\|_2$; $\phi_i := \|q_{i-1}\|_2$; $\psi_i := \|z_{i-1}\|_2$; $\nu_i := \|m_{i-1}\|_2$
    end if
    $m_i := M^{-1}w_i$
    $v_i := Am_i$
    if $i > 0$ **then**
      $\beta_i := \gamma_i/\gamma_{i-1}$; $\alpha_i := (\delta/\gamma_i - \beta_i/\alpha_{i-1})^{-1}$
    else
      $\beta_i := 0$; $\alpha_i := \gamma_i/\delta$
    end if
    $z_i := v_i + \beta_i z_{i-1}$
    $q_i := m_i + \beta_i q_{i-1}$
    $s_i := w_i + \beta_i s_{i-1}$
    $p_i := u_i + \beta_i p_{i-1}$
    $x_{i+1} := x_i + \alpha_i p_i$
    $r_{i+1} := r_i - \alpha_i s_i$
    $u_{i+1} := u_i - \alpha_i q_i$
    $w_{i+1} := w_i - \alpha_i z_i$
    if $i > 0$ **then**
      $e^f_{i-1} := \theta\chi_i + 2|\alpha_{i-1}|\,\theta\pi_i + \rho_i + 2|\alpha_{i-1}|\,\sigma_i$
      $e^h_{i-1} := \theta\xi_i + 2|\alpha_{i-1}|\,\theta\phi_i + \omega_i + 2|\alpha_{i-1}|\,\psi_i$
      if $i > 1$ **then**
        $e^g_{i-1} := \theta\xi_i + 2|\beta_{i-1}|\,\theta\pi_{i-1} + \omega_i + 2|\beta_{i-1}|\,\sigma_{i-1}$
        $e^j_{i-1} := (\mu\sqrt{n} + 2)\theta\nu_i + 2|\beta_{i-1}|\,\theta\phi_{i-1} + 2|\beta_{i-1}|\,\psi_{i-1}$
      end if
      if $i = 1$ **or** replace := true **then**
        $f_i := \epsilon\sqrt{(\mu\sqrt{n}+1)\theta\chi_i + \zeta} + \epsilon\sqrt{|\alpha_{i-1}|\,\mu\sqrt{n}\theta\pi_i} + \sqrt{e^f_{i-1}}\epsilon$
        $g_{i-1} := \epsilon\sqrt{\mu\sqrt{n}\theta\pi_i}$
        $h_i := \epsilon\sqrt{\mu\sqrt{n}\theta\xi_i} + \epsilon\sqrt{|\alpha_{i-1}|\,\mu\sqrt{n}\theta\phi_i} + \sqrt{e^h_{i-1}}\epsilon$
        $j_{i-1} := \epsilon\sqrt{\mu\sqrt{n}\theta\phi_i}$
        replace := false
      else
        $f_i := f_{i-1} + |\alpha_{i-1}|\,|\beta_{i-1}|\,g_{i-2} + |\alpha_{i-1}|\,h_{i-1} + \sqrt{e^f_{i-1}}\epsilon + |\alpha_{i-1}|\,\sqrt{e^g_{i-1}}\epsilon$
        $g_{i-1} := |\beta_{i-1}|\,g_{i-2} + h_{i-1} + \sqrt{e^g_{i-1}}\epsilon$
        $h_i := h_{i-1} + |\alpha_{i-1}|\,|\beta_{i-1}|\,j_{i-2} + \sqrt{e^h_{i-1}}\epsilon + |\alpha_{i-1}|\,\sqrt{e^j_{i-1}}\epsilon$
        $j_{i-1} := |\beta_{i-1}|\,j_{i-2} + \sqrt{e^j_{i-1}}\epsilon$
      end if
      if $f_{i-1} \le \tau\rho_i$ **and** $f_i > \tau\rho_{i+1}$ **then**
        $s_i := Ap_i$; $q_i := M^{-1}s_i$; $z_i := Aq_i$
        $r_{i+1} := b - Ax_{i+1}$; $u_{i+1} := M^{-1}r_{i+1}$; $w_{i+1} := Au_{i+1}$
        replace := true
      end if
    end if
  end for
end procedure

TABLE 1

Model problem 2D Laplacian operators of various sizes. A linear system with right-hand side $b = A\hat{x}$, where $\hat{x}_j = 1/\sqrt{n}$ is solved with the four presented algorithms. The initial guess is all-zero $\bar{x}_0 = 0$ (top table) and $\bar{x}_0 = rand(n, 1)$ (bottom table). The number of iterations required to reach maximal attainable accuracy is given, along with the corresponding relative true residual norm $\|b - A\bar{x}_i\|_2/\|b\|_2$ and the relative error (A-norm) $\|\hat{x} - \bar{x}_i\|_A/\|\hat{x}\|_A$. For the p-CG-rr method the number of replacement steps rr is indicated.

$\bar{x}_0 = 0$		CG		CG-CG		p-CG		p-CG-rr		
Matrix	n	iter	relres relerr	iter	relres relerr	iter	relres relerr	iter	relres relerr	rr
lapl50	2,500	128	7.8e-15 6.4e-15	127	8.1e-15 5.7e-15	118	1.5e-12 1.1e-12	125	9.1e-15 2.9e-14	3
lapl100	10,000	254	1.6e-14 1.4e-14	256	1.6e-14 1.4e-14	228	9.1e-12 6.5e-12	272	1.2e-14 1.8e-14	6
lapl200	40,000	490	3.1e-14 3.7e-14	487	3.2e-14 3.6e-14	439	5.4e-11 5.3e-11	536	2.5e-14 3.7e-14	11
lapl400	160,000	959	6.2e-14 1.0e-13	958	6.4e-14 5.6e-14	807	3.0e-10 3.4e-10	957	4.6e-14 1.8e-13	23
lapl800	640,000	1883	1.2e-13 2.7e-13	1877	1.3e-13 8.2e-13	1524	1.4e-10 2.0e-09	1876	1.1e-13 2.1e-13	53
$\bar{x}_0 = \text{rand}(n, 1)$		CG		CG-CG		p-CG		p-CG-rr		
Matrix	n	iter	relres relerr	iter	relres relerr	iter	relres relerr	iter	relres relerr	rr
lapl50	2,500	232	9.0e-14 4.9e-14	237	1.1e-13 6.4e-14	197	1.3e-10 1.1e-10	227	1.9e-14 7.4e-14	6
lapl100	10,000	444	2.9e-13 1.6e-13	449	3.8e-13 2.1e-13	367	1.5e-09 1.3e-09	483	1.6e-14 1.5e-14	10
lapl200	40,000	881	1.3e-12 6.2e-13	883	1.6e-12 7.8e-13	685	1.7e-08 1.6e-08	952	3.3e-14 3.9e-14	20
lapl400	160,000	1676	4.9e-12 2.3e-12	1714	6.1e-12 2.9e-12	1220	1.8e-07 1.8e-07	1846	1.1e-13 1.5e-13	35
lapl800	640,000	3339	2.1e-11 9.6e-12	3249	2.5e-11 1.2e-11	2225	1.9e-06 2.1e-06	3435	2.1e-12 2.4e-12	65

4. Numerical results. This section presents numerical results on a wide range of matrices to compare the behavior of the different CG methods and show the improved attainable accuracy using the automated residual replacement strategy. The numerical results in sections 4.1 and 4.2 are based on a MATLAB implementation of the different CG algorithms and their respective error estimates. Parallel performance measurements in section 4.3 result from a PETSc [1] implementation of p-CG-rr on a distributed memory machine using the message-passing paradigm.

4.1. Poisson model problem. The methods presented above are tested on a two-dimensional (2D) Laplacian PDE model with homogeneous Dirichlet boundary conditions, discretized using second-order finite differences on a uniform $n = n_x \times n_y$ point discretization of the unit square. The Poisson problem forms the basis for many practical HPC applications to which the pipelined CG method can be applied. Due to the very nature of the Laplace operator's spectrum, the application of CG to the Poisson problem typically does not display delayed convergence [26, 45]; see [20] for more details. This allows us to focus on the issue of reduced attainable accuracy that is observed when applying p-CG to large scale Poisson problems in this test case.

Table 1 shows convergence results for solving the discrete Poisson system for $n_x = n_y = 50, 100, 200, 400$, and 800, with condition numbers ranging from 1.5e+3 to 1.8e+5. A linear system with exact solution $\hat{x}_j = 1/\sqrt{n}$ (such that $\|\hat{x}\| = 1$) and right-hand side $b = A\hat{x}$ is solved for each of these discretization matrices. The initial guess is $\bar{x}_0 = 0$ (top table) and $\bar{x}_0 = \text{rand}(n, 1) \sim U([0, 1])$ (bottom table), respectively.

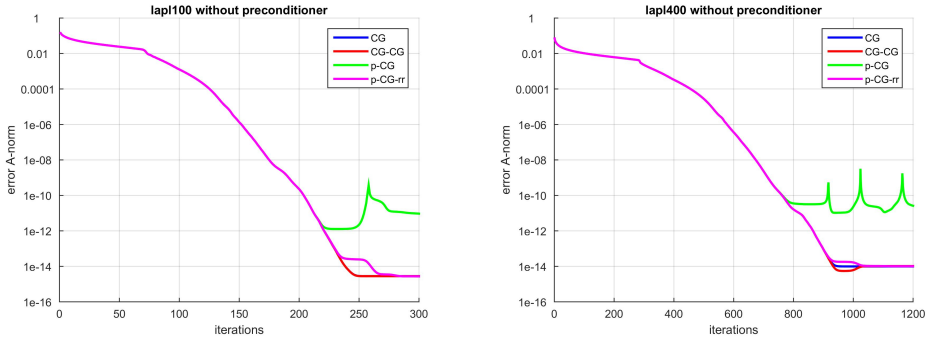

FIG. 1. *Error history for the different CG methods applied to the* `lapl`100 *(left) and* `lapl`400 *(right) matrices; see Table* 1. *Error A-norm* $\|\hat{x} - \bar{x}_i\|_A$ *as a function of iterations for CG (blue), Chronopoulos/Gear CG (red), pipelined CG (green), and p-CG-rr (magenta).*

The iteration is stopped when maximal attainable accuracy, i.e., $\min_i \|b - A\bar{x}_i\|_2$, is reached. This implies that a different stopping tolerance is used for each matrix. No preconditioner is used for the Laplace problems. The table lists the required number of iterations *iter*, the final relative true residual norm *relres*, and the final relative error *relerr* for the CG, CG-CG, p-CG, and p-CG-rr methods. Pipelined CG stagnates at a significantly larger residual and error compared to CG and CG-CG; see section 2. Note that for larger systems the loss of accuracy is dramatically more pronounced.

Figure 1 shows the A-norm of the error as a function of iterations for the `lapl100` and `lapl400` problems from Table 1. The CG method minimizes this quantity over the respective Krylov subspace in each iteration, which (in exact arithmetic) results in a monotonically decreasing error norm. For the pipelined CG method, the error norm behaves similar to the CG method up to its stagnation point. Beyond this point the error norm is no longer guaranteed to decrease. Periodic replacement of the residual and auxiliary variables improves the attainable accuracy, as illustrated by the monotonically decreasing p-CG-rr errors. However, a slight delay of convergence [45] is observed for the p-CG-rr method compared to classical CG; see also Table 1. We discuss the effect of rounding errors on orthogonality and the resulting delay of convergence near the end of section 4.2. Since the error is in general unavailable in practice, the remaining experiments focus on the norm of the residual instead.

Figure 2 illustrates the residual convergence history and the corresponding gap between the explicit and recursive residual in the different algorithms for the `lapl50` matrix. The right-hand side is $b_j = 1/\sqrt{n}$ in this experiment. The residual norm history of CG and CG-CG is nearly indistinguishable; the norm of the true residual at stagnation is 2.4e-13 and 2.5e-13, respectively. The pipelined CG method suffers from the amplification of local rounding errors, leading to early stagnation of the residual norm at 1.6e-11. The residual replacement strategy reduces the accuracy loss with a residual norm of 2.1e-13, which is comparable to classical CG.

4.2. Problems from Matrix Market. Numerical results on various linear systems are presented to show the effectiveness of pipelined CG with automated residual replacements. Table 2 lists all real, nondiagonal and SPD matrices from Matrix Market,[2] with their condition number κ, number of rows n, and total number of nonzero elements $\#nnz$. We solve a linear system with exact solution $\hat{x}_j = 1/\sqrt{n}$ and

[2] http://math.nist.gov/MatrixMarket.

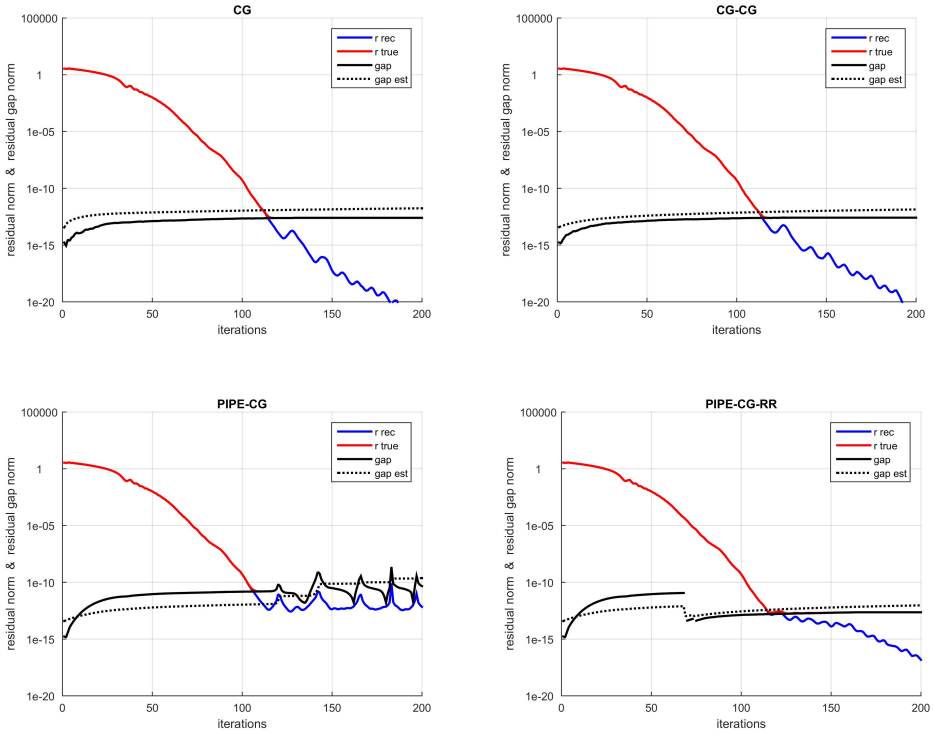

FIG. 2. *Residual norm history for the different CG methods applied to the* `lapl50` *matrix. Blue: recursive residual norms* $\|\bar{r}_i\|_2$. *Red: true residual norms* $\|b - A\bar{x}_i\|_2$. *Solid black lines: residual gap norms* $\|(b - A\bar{x}_i) - \bar{r}_i\|_2$ *(computed explicitly). Dotted black lines: residual gap estimates* $\|f_i\|_2$ *computed using the expression* (2.54).

right-hand side $b = A\hat{x}$ with the four presented methods, using an all-zero initial guess $\bar{x}_0 = 0$. Jacobi diagonal preconditioning (JAC) and Incomplete Cholesky (ICC) factorization are included to reduce the number of Krylov iterations if possible. For the preconditioners designated by *ICC an compensated Incomplete Cholesky factorization is performed, where a real nonnegative scalar η is used as a global diagonal shift in forming the Cholesky factor. For the `nos1` and `nos2` matrices the shift is $\eta = 0.5$, whereas for all other *ICC preconditioners we used $\eta = 0.1$.

Table 2 lists the number of iterations *iter* required to reach maximal accuracy and the corresponding explicitly computed relative residual norm *relres* for all methods. A "-" entry denotes failure to reach maximal accuracy within 5,000 iterations, in which case the relative residual after 5,000 iterations is displayed. The table indicates that for all test problems the residual replacement strategy incorporated in p-CG-rr improves the attainable accuracy of the p-CG method. For most matrices in the table the attainable accuracy is restored to the precision achieved by the classical CG method, although in some cases the increase in attainable accuracy is less pronounced.

Figure 3 illustrates the residual norm history for a few selected matrices from Table 2. The top panels show the true residual (solid) and residual gap (dashed) for the `bcsstk15` and `bcsstk18` matrices with Jacobi preconditioning. The bottom panels show the residual norm history for the `nos1` and `s1rmt3m1` matrices with ICC preconditioner. The residuals of the p-CG method level off sooner compared to classical CG and CG-CG. Based on the estimated residual gap (see (2.54)), the

TABLE 2

All real, non-diagonal and SPD matrices from Matrix Market, listed with their respective condition number $\kappa(A)$, number of rows/columns n, and total number of nonzeros $\#nnz$. A linear system with right-hand side $b = A\hat{x}$, where $\hat{x}_i = 1/\sqrt{n}$ is solved with each matrix with the four presented algorithms. The initial guess is all-zero $\bar{x}_0 = 0$. JAC and ICC preconditioners are included where needed. The number of iterations iter required to reach maximal attainable accuracy (stagnation point) and the corresponding relative true residuals relres $= \|b - A\bar{x}_i\|_2/\|b\|_2$ are shown. For the p-CG-rr method the number of replacement steps is indicated as rr.

Matrix	Prec	$\kappa(A)$	n	$\#nnz$	$\|b\|_2$	CG		CG-CG		p-CG		p-CG-rr	
						iter	relres	iter	relres	iter	relres	iter	relres	rr
bcsstk14	JAC	1.3e+10	1806	63,454	2.1e+09	650	7.6e-16	658	7.1e-16	506	5.2e-12	658	5.2e-16	9
bcsstk15	JAC	8.0e+09	3948	117,816	4.3e+08	772	3.7e-15	785	3.5e-15	646	2.3e-11	974	4.0e-15	10
bcsstk16	JAC	65	4884	290,378	1.5e+08	298	3.5e-15	300	4.0e-15	261	8.7e-12	301	2.1e-15	4
bcsstk17	JAC	65	10,974	428,650	9.0e+07	3547	1.0e-14	3428	1.7e-14	2913	2.8e-09	4508	1.2e-14	54
bcsstk18	JAC	65	11,948	149,090	2.6e+09	2299	2.2e-15	2294	2.1e-15	1590	2.9e-11	2400	1.5e-15	50
bcsstk27	JAC	7.7e+04	1224	56,126	1.1e+05	345	3.2e-15	345	4.0e-15	295	8.0e-12	342	2.7e-15	6
gr_30_30	-	3.8e+02	900	7744	1.1e+00	56	2.7e-15	55	3.1e-15	52	2.0e-13	61	3.0e-15	2
nos1	*ICC	2.5e+07	237	1017	5.7e+07	301	1.3e-14	338	1.2e-14	337	2.6e-10	968	1.9e-14	21
nos2	*ICC	6.3e+09	957	4137	1.8e+09	3180	8.3e-14	3292	1.1e-13	2656	1.2e-07	4429	2.7e-11	113
nos3	ICC	7.3e+04	960	15,844	1.0e+01	64	1.0e-14	63	1.1e-14	59	1.0e-12	61	2.5e-14	3
nos4	ICC	2.7e+03	100	594	5.2e-02	31	1.9e-15	31	1.9e-15	29	4.0e-14	33	1.3e-15	2
nos5	ICC	2.9e+04	468	5172	2.8e+05	63	3.2e-16	64	3.4e-16	58	4.3e-14	65	2.3e-16	2
nos6	ICC	8.0e+06	675	3255	8.6e+04	34	5.1e-15	35	6.2e-15	31	5.5e-11	33	1.0e-14	2
nos7	ICC	4.1e+09	729	4617	8.6e+03	29	4.0e-14	31	2.8e-14	29	4.5e-14	29	3.0e-14	3
s1rmq4m1	ICC	1.8e+06	5489	262,411	1.5e+04	122	4.3e-15	122	4.6e-15	114	5.5e-12	135	3.7e-15	6
s1rmt3m1	ICC	2.5e+06	5489	217,651	1.5e+04	229	9.3e-15	228	8.7e-15	213	2.2e-11	240	1.7e-14	9
s2rmq4m1	*ICC	1.8e+08	5489	263,351	1.5e+03	370	6.7e-15	387	7.3e-15	333	2.7e-10	349	2.5e-13	25
s2rmt3m1	ICC	2.5e+08	5489	217,681	1.5e+03	285	8.7e-15	283	1.0e-14	250	7.3e-10	425	8.7e-15	17
s3dkq4m2	*ICC	1.9e+11	90,449	2,455,670	6.8e+01	-	1.9e-08	-	2.1e-08	-	2.8e-07	-	5.6e-08	199
s3dkt3m2	*ICC	3.6e+11	90,449	1,921,955	6.8e+01	-	2.9e-07	-	2.9e-07	-	3.5e-07	-	2.9e-07	252
s3rmq4m1	*ICC	1.8e+10	5489	262,943	1.5e+02	1651	1.5e-14	1789	1.6e-14	1716	2.6e-08	1602	5.3e-10	154
s3rmt3m1	*ICC	2.5e+10	5489	217,669	1.5e+02	2282	2.7e-14	2559	2.9e-14	2709	9.3e-08	3448	8.0e-10	149
s3rmt3m3	*ICC	2.4e+10	5357	207,123	1.3e+02	2862	3.3e-14	2798	3.4e-14	3378	2.0e-07	2556	7.1e-11	248

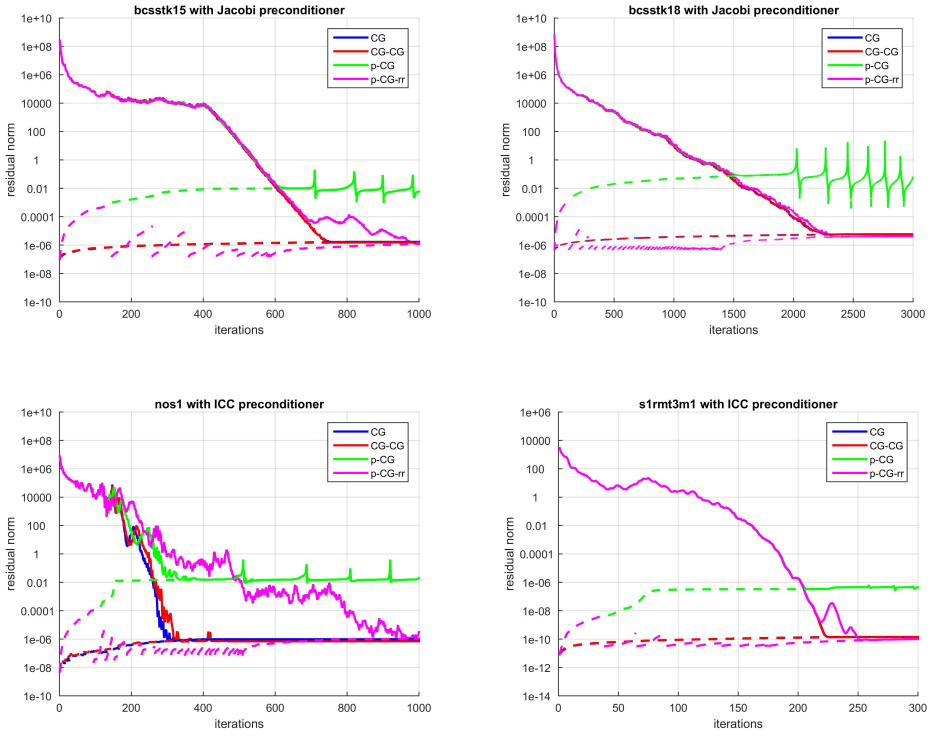

FIG. 3. *Residual norm history for the different CG methods applied to four SPD test matrices from Table 2. Solid lines: true residual norm $\|b - A\bar{x}_i\|_2$; dashed lines: residual gap $\|f_i\|_2$. Convergence of CG (blue) and Chronopoulos/Gear CG (red) is largely comparable. The pipelined CG method (green) suffers from rounding error propagation. Automated residual replacement (magenta) reduces the rounding errors, leading to an accuracy that is comparable to classical CG.*

residual replacement strategy explicitly computes the residual in the iterations where the criterion (3.5) is satisfied, leading to a more accurate final solution.

Note that the behavior of the p-CG and p-CG-rr residuals near the stagnation point differs slightly from the classical CG residuals. Furthermore, we point out that for the `nos1` matrix (see Figure 3 (bottom left)), as well as several other matrices from Tables 1 and 2, the p-CG and p-CG-rr methods show significantly delayed convergence. Indeed, apart from the loss of attainable accuracy, the propagation of local rounding errors in multiterm recurrence variants of iterative schemes can cause a convergence slowdown. We refer to [45, section 5] and the references therein, in particular [23], [26], and [34], for a more detailed discussion on delay of convergence by loss of orthogonality due to round-off. This delay translates into a larger number of iterations required to reach a certain accuracy; see Table 2. Although the residual replacement strategy ensures that a high accuracy can be obtained, it does not resolve the delay of convergence, as illustrated by the numerical results in this section. Hence, when application demands, a high accuracy can always be obtained using the p-CG-rr method, but this may come at the cost of additional iterations, inducing a trade-off between accuracy and computational effort.

4.3. Parallel performance. This section demonstrates that the parallel scalability of the pipelined CG method is maintained by the addition of the residual replacement strategy, and a significant speedup over classical CG can thus be obtained.

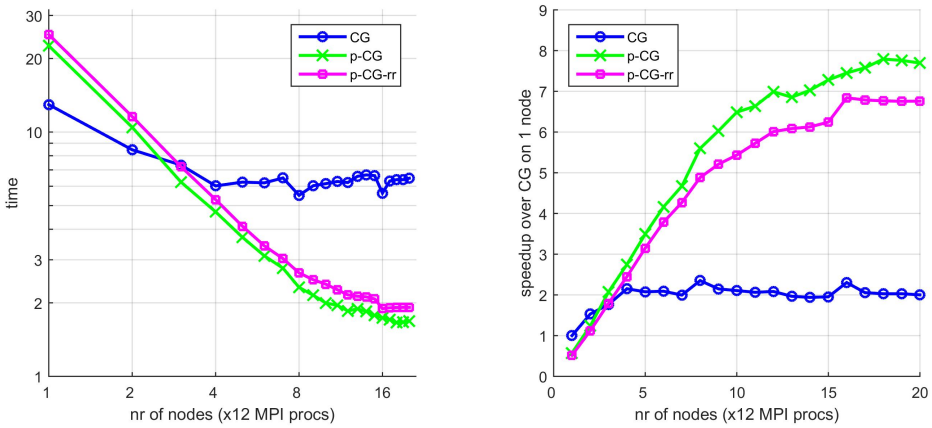

FIG. 4. *Strong scaling experiment on up to* 20 *nodes (*240 *cores) for a* 2*D Poisson problem with* 1,000,000 *unknowns. Left: absolute time to solution (*`log`10 *scale) as a function of the number of nodes (*`log`2 *scale). Right: speedup over single-node classical CG. All methods converged in* 1,474 *iterations to a relative residual tolerance* 1e-6*; p-CG-rr performed* 39 *replacements.*

The following parallel experiments are performed on a small cluster with 28 compute nodes, consisting of two 6-core Intel Xeon X5660 Nehalem 2.80-GHz processors each (12 cores per node). Nodes are connected by $4 \times$ QDR InfiniBand technology, providing 32 Gb/s of point-to-point bandwidth for message passing and I/O.

We use PETSc version 3.6.3. The benchmark problem used to assess strong scaling parallel performance is a moderately sized 2D Poisson model, available in the PETSc distribution as example 2 in the Krylov subspace solvers (KSP) folder. The simulation domain is discretized using a second-order finite difference stencil with $1{,}000 \times 1{,}000$ grid points (1 million unknowns). No preconditioner is applied. The tolerance imposed on the scaled recursive residual norm $\|\bar{r}_i\|_2/\|b\|_2$ is 10^{-6}. Since each node consists of 12 cores, we use 12 MPI processes per node to fully exploit parallelism on the machine. The MPI library used for this experiment is MPICH-3.1.3.[3] Note that the environment variables `MPICH_ASYNC_PROGRESS=1` and `MPICH_MAX_THREAD_SAFETY=multiple` are set to ensure optimal parallelism by allowing for nonblocking global communication.

Figure 4 (left) shows the time to solution as a function of the number of nodes (strong scaling). In this benchmark problem, pipelined CG starts to outperform classical CG when the number of nodes exceeds 2. Classical CG stops scaling from 4 nodes onward due to communication overhead. The pipelined methods scale well on up to 20 nodes for this problem; see Figure 4 (right). The maximum speedup for p-CG on 20 nodes compared to CG on a single node is 7.7×, whereas the CG method achieves a speedup of only 2.0× on 20 nodes. This implies pipelined CG attains a net speedup of 3.8× over classical CG for the current benchmark problems when both are executed on 20 nodes.[4] Performance of p-CG-rr is comparable to that of p-CG.

[3] http://www.mpich.org.

[4] The theoretical time per iteration (tpi) of CG, algorithm 1, is $2G + S$, where G is the tpi spent by the global communication phase and S is the tpi for the SPMV. The tpi for p-CG, Algorithm 3, is $\max(G, S)$; see [22, section 5]. However, a third dot-product is computed in the PETSc implementations of CG and p-CG to compute the norm $\|\bar{r}_i\|_2$. The tpi for PETSc's CG is thus $3G + S$. For p-CG this extra dot-product is combined into the existing global reduction phase such that the tpi remains unaltered. Hence, when $G = S$, a theoretical maximal speedup factor of 4× can be achieved by p-CG.

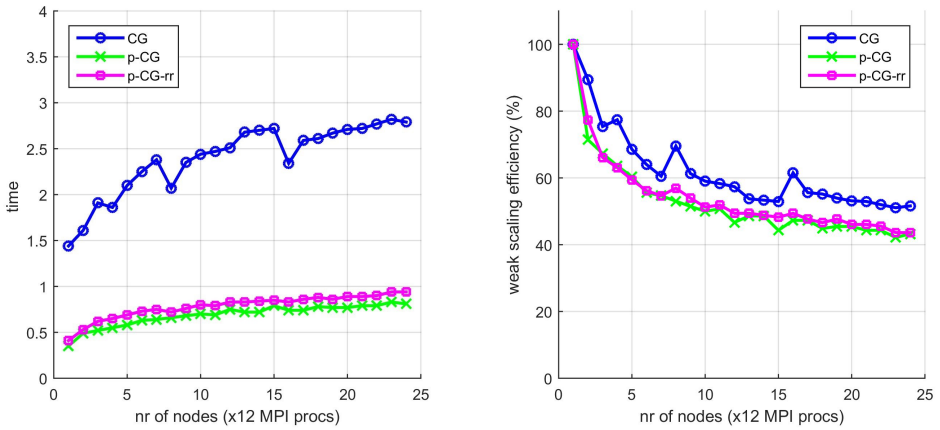

FIG. 5. *Weak scaling experiment on up to* 24 *nodes (*288 *cores) for a* 2*D Poisson problem with* $62,500$ *unknowns per node (*5200 *unknowns/core). Left: absolute time* t_{CG} *(*600 *iterations) as a function of the number of nodes. Right: weak scaling efficiency relative to single-node execution:* $eff_{CG}(m) = t_{CG}(1\,node)/t_{CG}(m\,nodes)$. *p-CG-rr performed* 10 *replacements.*

The minor observed slowdown is primarily due to the additional computational work required for the SPMVs (3.1) when replacement takes place. The p-CG-rr algorithm achieves a speedup of 3.4× over CG on 20 nodes for this problem and hardware setup. Note that the pipelined variants are effectively slower than classical CG on one or two nodes. This is due to the additional AXPYs in the pipelined methods, which require a significant amount of time for smaller numbers of nodes but are negligible on large numbers of nodes due to parallelism. This observation illustrates that good parallel algorithms are not necessarily the best sequential ones.

Figure 5 displays results for a weak scaling experiment, where the size of the Poisson problem grows linearly with respect to the number of cores. A fixed number of $62,500$ unknowns per node is used. The problem hence consists of $1{,}225 \times 1{,}225$ (1.5 million) unknowns on 24 nodes. Figure 5 (left) shows the time required to perform 600 iterations (fixed) of the various methods on up to 24 nodes. The speedup observed for the pipelined methods in Figure 4 is again visible here. The weak scaling efficiency of the p-CG and p-CG-rr algorithms (relative to their respective single-node execution) on 24 nodes (43%) is comparable to that of classical CG (51%); see Figure 5 (right).

Figure 6 shows the accuracy of the solution as a function of the number of iterations (left) and computational time (right) spent by the algorithms for the 2D Poisson $1{,}000 \times 1{,}000$ benchmark problem on a 20 node setup. In 3.2 seconds (∼2,500 iterations) the p-CG-rr algorithm obtains a solution with true residual norm 7.5e-12. Classical CG is over three times slower, requiring 11.1 seconds to attain a comparable accuracy (residual norm 9.4e-12); see also Figure 4. The p-CG method without residual replacement is unable to reach a comparable accuracy regardless of computational effort. Indeed, stagnation of the true residual norm around 2.0e-7 is imminent from a total time of 2.0 seconds (∼1,800 iterations) onward. For completeness we note that the speedup of p-CG/p-CG-rr over classical CG can also be obtained for less accurate final solutions, e.g., with $\|\bar{r}_i\| = 10^{-8}$ or 10^{-6}, as shown by Figure 6 (right).

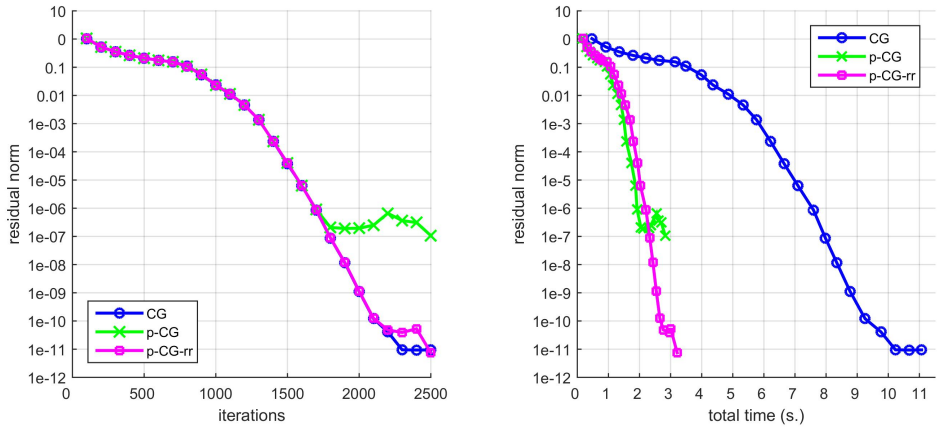

FIG. 6. *Accuracy experiment on* 20 *nodes (*240 *cores) for a* 2*D Poisson problem with* 1,000,000 *unknowns. Left: explicitly computed residual as a function of iterations. Right: residual as a function of total time spent by the algorithm. Maximal number of iterations is* 2,500 *for all methods; p-CG-rr performed (maximum)* 39 *replacements.*

5. Conclusions. Deviation of the recursively computed residuals from the true residuals due to the propagation of local rounding errors is a well-known issue in many numerical methods. This behavior is significantly more prominent in multiterm recursion variants of CG, such as the three-term recurrence CG algorithm [43], Chronopoulos and Gear's communication-avoiding CG (CG-CG) [6], and the communication-hiding pipelined CG method (p-CG) [22]. For these methods, the dramatic amplification of local rounding errors may lead to a stagnation of the residual norm at several orders of magnitude above the accuracy attainable by classical CG.

This paper aims to lay the foundation for the analysis of the propagation of local rounding errors that stem from the recursions in classical CG, CG-CG, and the pipelined CG algorithm. Pipelined CG features additional recursively computed auxiliary variables compared to classical CG which are all prone to rounding errors. We show that the gap between the explicitly computed and recursive residual is directly related to the gaps on the other recursively defined auxiliary variables. A bound on the residual gap norm is derived which provides insight into the observed accuracy loss in the pipelined CG algorithm. Furthermore, a practically usable estimate for the residual gap is suggested. Based on this estimate, a heuristic to compensate for the loss of attainable accuracy is proposed in the form of an automated residual replacement strategy. However, since the assumption of Krylov basis orthogonality is not guaranteed to hold in finite precision, the replacement strategy should be interpreted as an effective yet primarily intuitive practical tool to improve attainable accuracy.

The residual replacement strategy is illustrated on a variety of numerical benchmark problems. For most test problems the replacements allow to attain a significantly improved accuracy which is unobtainable by pipelined CG. However, a delay of convergence due to the replacements is observed for specific problems. Although the incorporation of the replacement strategy in the algorithm requires the calculation of several additional vector norms, these computations can easily be combined with the existing global communication phase. Performance results with a parallel implementation of

p-CG-rr in PETSc using the MPI message-passing paradigm indicate that the replacement strategy improves accuracy but does not impair parallel performance.

While this work aims to be a contribution towards more efficient and more accurate parallel variants of the CG algorithm, we point out that many open questions in this area still remain. Future work may (and should) tackle the issues of effectively accounting for the loss of orthogonality in finite precision in the numerical framework presented in this paper as well as the analysis and possible remedy for the observed delay of convergence in multiterm recurrence CG variants, including pipelined CG.

Acknowledgments. The authors would like to cordially thank both Zdeněk Strakoš and the anonymous SIMAX referee for their useful comments and valuable suggestions on earlier versions of this manuscript.

REFERENCES

[1] S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, L. C. McInnes, K. Rupp, B. Smith, S. Zampini, and H. Zhang, *PETSc Web page*, http://www.mcs.anl.gov/petsc, 2015.

[2] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 2nd ed., SIAM, Philadelphia, PA, 1994.

[3] E. Carson and J. Demmel, *A residual replacement strategy for improving the maximum attainable accuracy of s-step Krylov subspace methods*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 22–43.

[4] E. Carson, N. Knight, and J. Demmel, *Avoiding communication in nonsymmetric Lanczos-based Krylov subspace methods*, SIAM J. Sci. Comput., 35 (2013), pp. S42–S61.

[5] E. Carson, M. Rozložník, Z. Strakoš, P. Tichý, and M. Tůma, *On the numerical stability analysis of pipelined Krylov subspace methods*, SIAM J. Sci. Comput., submitted.

[6] A. Chronopoulos and C. Gear, *s-Step iterative methods for symmetric linear systems*, J. Comput. Appl. Math., 25 (1989), pp. 153–168.

[7] A. Chronopoulos and A. Kucherov, *Block s-step Krylov iterative methods*, Numer. Linear Algebra Appl., 17 (2010), pp. 3–15.

[8] A. Chronopoulos and C. Swanson, *Parallel iterative s-step methods for unsymmetric linear systems*, Parallel Comput., 22 (1996), pp. 623–641.

[9] E. D'Azevedo, V. Eijkhout, and C. Romine, *Reducing Communication Costs in the Conjugate Gradient Algorithm on Distributed Memory Multiprocessors*, Technical report TM/12192, Oak Ridge National Lab, Oak Ridge, TN, 1992.

[10] E. de Sturler, *A parallel variant of GMRES(m)*, in Proceedings of the 13th IMACS World Congress on Computational and Applied Mathematics, Vol. 9, Dublin, Ireland, Criterion Press, 1991, pp. 682–683.

[11] E. De Sturler and H. Van der Vorst, *Reducing the effect of global communication in GMRES(m) and CG on parallel distributed memory computers*, Appl. Numer. Math., 18 (1995), pp. 441–459.

[12] J. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.

[13] J. Demmel, M. Heath, and H. Van der Vorst, *Parallel Numerical Linear Algebra*, Acta Numer., 2 (1993), pp. 111–197.

[14] J. Dongarra, P. Beckman, T. Moore, P. Aerts, G. Aloisio, J. Andre, D. Barkai, J. Berthou, T. Boku, B. Braunschweig, et al., *The international exascale software project roadmap*, Internat. J. High Performance Comput. Appli., 25 (2011), pp. 3–60.

[15] J. Dongarra, I. Duff, D. Sorensen, and H. Van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, SIAM, Philadelphia, PA, 1998.

[16] J. Dongarra and M. Heroux, *Toward a New Metric for Ranking High Performance Computing Systems*, Technical report SAND2013-4744, 312, Sandia National Laboratories, Livermore, CA, 2013.

[17] J. Dongarra, M. Heroux, and P. Luszczek, *HPCG Benchmark: A New Metric for Ranking High Performance Computing Systems*, Knoxville, technical report UT-EECS-15-736, University of Tennessee, 2015.

[18] P. Eller and W. Gropp, *Scalable non-blocking preconditioned conjugate gradient methods*, in SC16: International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE, 2016, pp. 204–215.

[19] J. Erhel, *A parallel GMRES version for general sparse matrices*, Electron. Trans. Numer. Anal., 3 (1995), pp. 160–176.

[20] T. Gergelits and Z. Strakoš, *Composite convergence bounds based on Chebyshev polynomials and finite precision conjugate gradient computations*, Numer. Algorithms, 65 (2014), pp. 759–782.
[21] P. Ghysels, T. Ashby, K. Meerbergen, and W. Vanroose, *Hiding global communication latency in the GMRES algorithm on massively parallel machines*, SIAM J. Sci. Comput., 35 (2013), pp. C48–C71.
[22] P. Ghysels and W. Vanroose, *Hiding global synchronization latency in the preconditioned conjugate gradient algorithm*, Parallel Comput., 40 (2014), pp. 224–238.
[23] A. Greenbaum, *Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences*, Linear Algebra Appl., 113 (1989), pp. 7–63.
[24] A. Greenbaum, *Estimating the attainable accuracy of recursively computed residual methods*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 535–551.
[25] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia, PA, 1997.
[26] A. Greenbaum and Z. Strakoš, *Predicting the behavior of finite precision Lanczos and conjugate gradient computations*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 121–137.
[27] M. Gutknecht and Z. Strakoš, *Accuracy of two three-term and three two-term recurrences for Krylov space solvers*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 213–229.
[28] N. Halko, P. Martinsson, and J. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288.
[29] M. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Natl. Bureau Standards, 49 (1952), pp. 409–436.
[30] N. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadelphia, PA, 2002.
[31] J. Liesen and Z. Strakoš, *Krylov Subspace Methods: Principles and Analysis*, Oxford University Press, Oxford, 2012.
[32] G. Meurant, *Computer Solution of Large Linear Systems*, Vol. 28, Elsevier, Amsterdam, 1999.
[33] G. Meurant and Z. Strakoš, *The Lanczos and conjugate gradient algorithms in finite precision arithmetic*, Acta Numer., 15 (2006), pp. 471–542.
[34] Y. Notay, *On the convergence rate of the conjugate gradients in presence of rounding errors*, Numer. Math., 65 (1993), pp. 301–317.
[35] C. Paige, *The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices*, Ph.D. thesis, University of London, London, 1971.
[36] C. Paige, *Computational variants of the Lanczos method for the eigenproblem*, IMA J. Appl. Math., 10 (1972), pp. 373–381.
[37] C. Paige, *Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix*, IMA J. Appl. Math., 18 (1976), pp. 341–349.
[38] C. Paige, *Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem*, Linear Algebra Appl., 34 (1980), pp. 235–258.
[39] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, 2003.
[40] G. Sleijpen and H. Van der Vorst, *Reliable updated residuals in hybrid Bi-CG methods*, Computing, 56 (1996), pp. 141–163.
[41] G. Sleijpen, H. Van der Vorst, and D. Fokkema, *BiCGstab(ℓ) and other hybrid Bi-CG methods*, Numer. Algorithms, 7 (1994), pp. 75–109.
[42] G. Sleijpen, H. Van der Vorst, and J. Modersitzki, *Differences in the effects of rounding errors in Krylov solvers for symmetric indefinite linear systems*, SIAM J. Matrix Anal. Appl., 22 (2001), pp. 726–751.
[43] E. Stiefel, *Relaxationsmethoden bester Strategie zur Lösung linearer Gleichungssysteme*, Comment. Math. Helv., 29 (1955), pp. 157–179.
[44] Z. Strakoš, *Effectivity and optimizing of algorithms and programs on the host-computer/array-processor system*, Parallel Comput., 4 (1987), pp. 189–207.
[45] Z. Strakoš and P. Tichý, *On error estimation in the Conjugate Gradient method and why it works in finite precision computations*, Electron. Trans. Numer. Anal., 13 (2002), pp. 56–80.
[46] Z. Strakoš and P. Tichý, *Error estimation in preconditioned Conjugate Gradients*, BIT Numer. Math., 45 (2005), pp. 789–817.
[47] C. Tong and Q. Ye, *Analysis of the finite precision Bi-Conjugate Gradient algorithm for nonsymmetric linear systems*, Math. Comp., 69 (2000), pp. 1559–1575.
[48] H. Van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Vol. 13, Cambridge University Press, Cambridge, 2003.
[49] H. Van der Vorst and Q. Ye, *Residual replacement strategies for Krylov subspace iterative methods for the convergence of true residuals*, SIAM J. Sci. Comput., 22 (2000), pp. 835–852.
[50] J. Wilkinson, *Rounding Errors in Algebraic Processes*, Courier Corporation, North Chelms ford, MA, 1994.