A Residual Replacement Strategy for Improving the Maximum Attainable Accuracy of s-step Krylov Subspace Methods

Erin Carson
James Demmel

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2012-197
http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-197.html

September 19, 2012

Acknowledgement

This paper was researched with Government support under and awarded by the Department of Defense, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a. This research was also supported by Microsoft (Award #024263) and Intel (Award #024894) funding and by matching funding by U.C. Discovery (Award #DIG07-10227). Additional support comes from Par Lab affiliates National Instruments, Nokia, NVIDIA, Oracle, and Samsung, U.S. DOE grants DE-SC0003959, DE-AC02-05-CH11231, Lawrence Berkeley National Laboratory, and NSF SDCI under Grant Number OCI-1032639.

A RESIDUAL REPLACEMENT STRATEGY FOR IMPROVING THE MAXIMUM ATTAINABLE ACCURACY OF S-STEP KRYLOV SUBSPACE METHODS

ERIN CARSON AND JAMES DEMMEL

Abstract. Recent results have demonstrated the performance benefits of communication-avoiding Krylov subspace methods, variants which use blocking strategies to perform $O(s)$ computation steps of the algorithm for each communication step. This allows an $O(s)$ reduction in total communication cost, which can lead to significant speedups on modern computer architectures. Despite potential performance benefits, stability of these variants has been an obstacle to their use in practice.

Following equivalent analyses for the classical Krylov methods, we bound the deviation of the true and computed residual in finite precision communication-avoiding Krylov methods, which leads to an implicit residual replacement strategy. We are the first, to our knowledge, to perform this analysis for s-step variants. We show that our strategy can be implemented without affecting the asymptotic communication or computation cost of the algorithm, for both the sequential and parallel case. Numerical experiments demonstrate the effectiveness of our residual replacement strategy for communication-avoiding variants of both the conjugate gradient and biconjugate gradient methods. Specifically, it is shown that for cases where the computed residual converges, accuracy of order $O(\epsilon)\,\|A\|\,\|x\|$ can be achieved with a small number of residual replacement steps, demonstrating the potential for practical use of communication-avoiding Krylov subspace methods.

Key words. Krylov subspace methods, maximum attainable accuracy, residual replacement, numerical stability, iterative solvers, minimizing communication, sparse matrix

AMS subject classifications. 65N12, 65N15, 65F10, 65F50, 65Y05, 68W40, 65G50

1. Introduction.

Krylov subspace methods (KSMs) are a class of iterative algorithms commonly used to solve linear systems. These methods work by iteratively adding a dimension to a Krylov subspace and then choosing the "best" solution from the resulting subspace, where the solution x^k and residual r^k are updated as

$$x^k = x^{k-1} + \alpha_{k-1}p^{k-1} \quad r^k = r^{k-1} - \alpha_{k-1}Ap^{k-1} \tag{1.1}$$

or something similar. Although the above formula applies specifically to conjugate gradient (CG) and biconjugate gradient (BICG), similar expressions describe steepest descent, conjugate gradient squared (CGS), stabilized biconjugate gradient (BICGSTAB), and other recursively computed residual methods.

It is important to notice that x^k and r^k have different round-off patterns. That is, the expression for x^k does not depend on r^k, nor does the expression for r^k depend on x^k. Therefore, computational errors made in x^k are not self-correcting. Throughout the iteration, these errors accumulate and cause deviation of the true residual, $b - Ax^k$, and computed residual, r^k. This limits the *maximum attainable accuracy*, which bounds how accurately we can solve the system on a computer with machine precision ϵ. When maximum attainable accuracy is attained, the computed residual will continue decreasing in norm, whereas the norm of the true residual will stagnate. It is therefore possible to have very large error in the solution despite the algorithm reporting a very small residual norm.

Strategies such as restarting and residual replacement are commonly used to limit the error that accumulates throughout the computation (see, e.g., [20, 24]). Simply substituting the true residual for the Lanczos recurrence in each iteration (or even every s iterations), in addition to increasing both the communication and computation

in the method, can destroy the super-linear convergence properties driven by the Lanczos process [20, 24]. Residual replacement strategies must therefore carefully select iterations where residual replacement takes place, which requires estimating the accrued rounding error. Van der Vorst and Ye have successfully implemented such a strategy for standard Krylov methods [24].

The computation involved in standard Krylov methods, namely, the updates of x^k and r^k, consists of one or more sparse matrix-vector multiplications (SpMVs) and vector operations in each iteration. Because these kernels have a low computation to communication ratio, standard Krylov method implementations are communication-bound on modern computer architectures.

This motivated $s-$step, or, Communication-Avoiding KSMs (CA-KSMs), which are equivalent to the standard KSM implementations in exact arithmetic. These variants use blocking strategies to perform $O(s)$ computation steps of the algorithm for each communication step, allowing an $O(s)$ reduction in total communication cost (see, e.g., [2, 3, 5, 7, 8, 10, 11, 16, 25, 21, 22]).

Despite attractive performance benefits, the effect of finite precision error in CA-KSMs is not well understood. The deviation of the true and computed residual, as well as the convergence rate, potentially become worse as s increases. Many previous authors have observed this behavior in CA-KSMs (see, e.g. [3, 4, 5, 11, 25, 26]). We are the first, to our knowledge, to provide a quantitative analysis of round-off error in these algorithms which limits the maximum attainable accuracy. Our analysis, which follows the analysis for classical KSMs in [24], leads directly to an implicit residual replacement strategy to reduce such error.

In all numerical experiments solving $Ax = b$, if the norm of the computed residual for the CA-KSM converges to $O(\epsilon) \|A\| \|x\|$, our residual replacement strategy is effective at maintaining agreement between residuals such that the true residual will also converge with norm equal to $O(\epsilon) \|A\| \|x\|$. Furthermore, in all tests, the number of residual replacement steps was small compared to the total number of iterations. The highest ratio of replacement steps to total iterations was 0.03; in most cases the ratio was less than 0.01. If we generate the Krylov basis using properly chosen Newton or Chebyshev polynomials, the basis additionally remains well conditioned even for large s, leading to a convergence rate close to that of the classical implementation. We therefore conclude that, for many applications, CA-KSMs can solve $Ax = b$ as accurately as the classical method for a factor of $O(s)$ less communication.

2. Related work. We briefly discuss review work in the areas of s-step and CA-KSMs, as well as work related to the numerical analysis of classical KSMs.

2.1. s-step Krylov subspace methods. The first instance of an s-step method in the literature is Van Rosendale's conjugate gradient method [25]. Van Rosendale's implementation was motivated by exposing more parallelism using the PRAM model. Chronopoulous and Gear later created an $s-$step GMRES method with the goal of exposing more parallel optimizations [4]. Walker looked into s-step bases as a method for improving stability in GMRES by replacing the modified Gram-Schmidt orthogonalization process with Householder QR [26]. All these authors used the monomial basis, and found that convergence often could not be guaranteed for $s > 5$. It was later discovered that this behavior was due to the inherent instability of the monomial basis, which motivated research into the use of other bases for the Krylov subspace.

Hindmarsh and Walker used a scaled (normalized) monomial basis to improve convergence [10], but only saw minimal improvement. Joubert and Carey implemented a scaled and shifted Chebyshev basis which provided more accurate results [12]. Bai

et al. also saw improved convergence using a Newton basis [1]. Although successively scaling the basis vectors lowers the condition number of the basis matrix, hopefully yielding convergence closer to that of the standard method, this computation reintroduces the dependency we sought to remove, hindering communication-avoidance. Hoemmen resolves this using a novel matrix equilibration and balancing approach as a preprocessing step, eliminating the need for scaled basis vectors [11].

Hoemmen et. al [7, 11, 16] have derived communication-avoiding variants of Lanczos, Arnoldi, CG and GMRES. The derivation of communication-avoiding variants of two-sided Krylov subspace methods, such as BICG, CGS, and BICGSTAB can be found in [2].

2.2. Error analysis of Krylov subspace methods. The behavior of classical Krylov subspace methods in finite precision arithmetic is a well-studied problem. The finite precision Lanczos process, which drives convergence, can lead to a significant deviation between the recursively computed residual and the true residual, $b - Ax^k$, decreasing the maximum attainable accuracy of the solution. Previous authors have devised residual replacement strategies for Krylov methods, in which the computed residual is replaced by the finite precision evaluation of the true residual at carefully chosen iterations, discussed in further detail below. In this way, agreement between the true and computed residual is maintained throughout the iteration.

An upper bound on the maximum attainable accuracy for KSMs was provided by Greenbaum [9]. Greenbaum proved that this bound can be given a priori for methods like CG, but can not be predetermined for methods like BICG, which can have arbitrarily large intermediate iterates. Additionally, Greenbaum has shown the backward stability of the CG algorithm, by showing that the Ritz values found lie in small intervals around the eigenvalues of A. There are many other analyses of the behavior of various KSMs in finite precision arithmetic (see, e.g. [15, 14, 23]). The reader is also directed to the bibliography in [18].

Sleijpen and Van der Vorst implemented a technique called "flying restarts" to decrease the amount of round-off error that occurs in KSMs [20]. Their method, which is applicable to many KSMs, iteratively tracks an upper bound for the amount of round-off that has occurred in the iteration so far. Using this upper bound, the algorithm may decide, at each iteration, to perform a *group update* (discussed in Sec. 4), to restart the algorithm (setting the right hand side appropriately), or both. The benefit from using a group update strategy is analogous to grouping to reduce round-off error in finite precision recursive summation. Following this work, Van der Vorst and Ye devised a residual replacement strategy, which, rather than restarting, replaces the residual with the computed value of the true residual, combined with group updates [24]. This residual replacement occurs at iterations chosen such that two objectives are met: first, the accumulated round-off does not grow so large as to limit the attainable accuracy, and second, the Lanczos process is not perturbed so much as to slow the rate of convergence. To determine when these conditions are met, the algorithm iteratively updates a bound on the error accrued thus far. The analysis for CA-KSMs in subsequent sections closely parallels the methodology of [24].

2.3. Communication-avoiding conjugate gradient. We briefly review the communication-avoiding conjugate gradient algorithm (CA-CG), shown in Alg.1. We chose CG for simplicity, although the same general techniques used here can be applied to other CA-KSMs as well. In the interest of space, we refer the reader to numerous other works on the topic, such as [3, 4, 7, 11, 13, 25, 22].

The CA-CG method has both an inner loop, which iterates from $j = 1 : s$, and k outer loop iterations, where k depends on the number of steps until convergence (or some other termination condition). Therefore, we will index quantities as $sk + j$ for clarity. For reference, Table 2.1 contains quantities used in the CA-CG derivation below along with their dimensions.

	A	P^k	R^k	V^k	V_j^k	G^k	$\bar{T}$	$\bar{T}_{j+1}$	T	T_{j+1}
Rows	N	N	N	N	N	$2s+1$	$s+1$	$j+1$	$2s+1$	$2j+1$
Cols	N	$s+1$	s	$2s+1$	$2j-1$	$2s+1$	s	j	$2s+1$	$2j+1$

TABLE 2.1
Dimensionality of variables in CA-CG derivation.

In CA-CG, we do not update x^{sk+j}, r^{sk+j}, and p^{sk+j} directly within the inner loop, but instead update their coefficients in the Krylov basis

$$V^k = [P^k, R^k] = [\rho_0(A)p^{sk}, ..., \rho_s(A)p^{sk}, \rho_0(A)r^{sk}, ..., \rho_{s-1}(A)r^{sk}] \tag{2.1}$$

where ρ_i is a polynomial of degree i. We assume a three-term recurrence for generating these polynomials, which can be written in terms of basis parameters γ_i, θ_i, and σ_i as

$$\rho_i(A) = \gamma_i(A - \theta_i I)\rho_{i-1}(A) + \sigma_i\rho_{i-2}(A) \tag{2.2}$$

This three-term recurrence covers a large class of polynomials (including all classical orthogonal polynomials). The monomial, Newton, and Chebyshev bases are common choices for CA-KSMs (see, e.g., [19]).

This Krylov basis is generated at the beginning of each outer loop, using the current r^{sk} and p^{sk} vectors, by the communication-avoiding matrix powers kernel described in [7, 11, 16]. We can then represent x^{sk+j}, r^{sk+j}, and p^{sk+j} by coefficients e^{sk+j}, c^{sk+j}, and a^{sk+j}, which are vectors of length $2s + 1$, such that

$$x^{sk+j} = V^k e^{sk+j} + x^{sk}, \quad r^{sk+j} = V^k c^{sk+j}, \quad p^{sk+j} = V^k a^{sk+j}$$

If we rearrange Eq. (2.2), we obtain an expression for multiplication by A as a linear combination of basis vectors, with weights defined in terms of the parameters γ_i, θ_i, and σ_i. In matrix form, this defines the tridiagonal matrix $\bar{T}$, of dimension $(s+1) \times s$, such that

$$AP_i^k = P_{i+1}^k \bar{T}_{i+1} \quad AR_i^k = R_{i+1}^k \bar{T}_{i+1}$$

where P_i^k, R_i^k are $N \times i$ matrices containing the first i columns of P^k or R^k, respectively, and $\bar{T}_{i+1}$ is the matrix containing the first $i + 1$ rows and first i columns of $\bar{T}$. Let $V_j^k = [P_j^k, R_{j-1}^k]$. This allows us to write

$$AV_j^k = A[P_j^k, R_{j-1}^k] = [P_{j+1}^k \bar{T}_{j+1}, R_j^k \bar{T}_j] = V_{j+1}^k \begin{bmatrix} \bar{T}_{j+1} & \\ & \bar{T}_j \end{bmatrix} \tag{2.3}$$

Let $T_{j+1} = \begin{bmatrix} [\bar{T}_{j+1}, 0_{s+1,1}] & \\ & [\bar{T}_j, 0_{s,1}] \end{bmatrix}$ and define $T \equiv T_{s+1}$. We can then write Ap^{sk+j} as

Algorithm 1 CA-CG Method

1: x^0, $r^0 = b - Ax^0$, $p^0 = r^0$, $k = 0$
2: **while** not converged **do**
3: Calculate P^k, R^k, T by (2.1) and (2.3)
4: Let $V^k = [P^k, R^k]$,$G^k = (V^k)^T V^k$
5: $a^{sk} = [1, 0_{2s}]^T$,$c^{sk} = [0_{s+1}, 1, 0_{s-1}]^T$, $e^{sk} = [0_{2s+1}]$
6: **for** $j = 1 : s$ **do**
7: $\alpha_{sk+j-1} = [(c^{sk+j-1})^T G^k (c^{sk+j-1})]/[(a^{sk+j-1})^T G^k (Ta^{sk+j-1})]$
8: $e^{sk+j} = e^{sk+j-1} + \alpha_{sk+j-1} a^{sk+j-1}$
9: $c^{sk+j} = c^{sk+j-1} - T\left(\alpha_{sk+j-1} a^{sk+j-1}\right)$
10: $\beta_{sk+j-1} = [(c^{sk+j})^T G^k (c^{sk+j})]/[(c^{sk+j-1})^T G^k (c^{sk+j-1})]$
11: $a^{sk+j} = c^{sk+j} + \beta_{sk+j-1} a^{sk+j-1}$
12: **end for**
13: $x^{sk+s} = [P^k, R^k]e^{sk+s} + x^{sk}$, $r^{sk+s} = [P^k, R^k]c^{sk+s}$, $p^{sk+s} = [P^k, R^k]a^{sk+s}$
14: $k = k + 1$
15: **end while**
16: **return** x^{sk}

$$\begin{aligned} Ap^{sk+j} = AV^k a^{sk+j} &= (AV_j^k) \begin{bmatrix} a_{0:j-1}^{sk+j} \\ a_{s+1:s+j-1}^{sk+j} \end{bmatrix} = V_{j+1}^k T_{j+1} \begin{bmatrix} a_{0:j}^{sk+j} \\ a_{s+1:s+j}^{sk+j} \end{bmatrix} \\ &= V^k T a^{sk+j} \end{aligned}$$

Therefore, Ta^{sk+j} are the Krylov basis coefficients for Ap^{sk+j}. This expression allows us to avoid explicit multiplication by A within the inner loop, and thus avoid communication.

3. Error in finite precision CA-KSMs . Throughout this analysis, we use a standard model of floating point arithmetic:

$$\begin{aligned} fl(x + y) &= x + y + \delta \quad \text{with } |\delta| \leq \epsilon(|x + y|) \\ fl(Ax) &= Ax + \delta \quad \text{with } |\delta| \leq \epsilon N_A |A| |x| \end{aligned}$$

where ϵ is the unit round-off of the machine, $x, y \in \mathbb{R}^N$, and N_A is a constant associated with the matrix-vector multiplication (for example, the maximal number of nonzero entries in a row of A). All absolute values and inequalities are componentwise. Using this model, we can also write

$$fl(y + Ax) = y + Ax + \delta \text{ with } |\delta| \leq \epsilon(|y + Ax| + N_A |A| |x|)$$

where, as in the remainder of the analysis, we ignore higher powers of ϵ for simplicity. We can now perform an analysis of round-off error in computing the updates in the CA-CG (and CA-BICG) method. Our goal is to bound the norm of the difference between the true and computed residual in terms of the norms of quantities which can be computed inexpensively in each iteration.

3.1. Error in finite precision iterate updates. First, we consider the finite precision error in updating e^{sk+j} and c^{sk+j} in the inner loop (lines 8 and 9 in Alg. 1) and performing the basis change to obtain x^{sk+s} and r^{sk+s} at the end of the s steps (line 13 in Alg. 1). In the communication-avoiding variant of CG, we represent the solution and computed residual by their coefficients in the Krylov basis $V^k = [P^k, R^k]$,

where P^k and R^k are the $O(s)$ dimensional Krylov bases with starting vectors p^{sk} and r^{sk}, respectively. These coefficient vectors are initialized as $e^{sk} = [0_{2s+1}]^T$, $c^{sk} = [0_{s+1}, 1, 0_{s-1}]^T$. In the inner loop, we update these coefficients as

$$e^{sk+j} = e^{sk+j-1} + \alpha_{sk+j-1} a^{sk+j-1} \tag{3.1}$$

$$c^{sk+j} = c^{sk+j-1} - T\left(\alpha_{sk+j-1} a^{sk+j-1}\right) \tag{3.2}$$

When (3.1) and (3.2) are implemented in finite precision, they become

$$\begin{aligned}
\hat{e}^{sk+j} &= fl(\hat{e}^{sk+j-1} + \alpha_{sk+j-1} a^{sk+j-1}) \\
&= \hat{e}^{sk+j-1} + \alpha_{sk+j-1} a^{sk+j-1} + \xi^{sk+j} && (3.3) \\
&\left|\xi^{sk+j}\right| \leq \epsilon \left|\hat{e}^{sk+j}\right| && (3.4) \\
\hat{c}^{sk+j} &= fl(\hat{c}^{sk+j-1} - T\left(\alpha_{sk+j-1} a^{sk+j-1}\right)) \\
&= \hat{c}^{sk+j-1} - T\left(\alpha_{sk+j-1} a^{sk+j-1}\right) + \eta^{sk+j} && (3.5) \\
&\left|\eta^{sk+j}\right| \leq \epsilon(\left|\hat{c}^{sk+j}\right| + N_T \left|T\right| \left|\alpha_{sk+j-1} a^{sk+j-1}\right|) && (3.6)
\end{aligned}$$

where we use hats to decorate values computed in finite precision. Note that the rounding errors in computing $\alpha_{sk+j-1} a^{sk+j-1}$ do not affect the numerical deviation of the true and computed residuals [9, 24]; the deviation of the two residuals is due to the different round-off patterns that come from different treatment of $\alpha_{sk+j-1} a^{sk+j-1}$ in the recurrences for e^{sk+j} and c^{sk+j}. We therefore let the term $\alpha_{sk+j-1} a^{sk+j-1}$ denote the computed quantity.

In the inner loop, we denote the exact expressions for the solution and computed residual represented in the Krylov basis by

$$x'^{sk+j} = V^k \hat{e}^{sk+j} + \hat{x}'^{sk} \tag{3.7}$$

$$r'^{sk+j} = V^k \hat{c}^{sk+j} \tag{3.8}$$

These expressions are computed in finite precision only at the end of each outer loop iteration, when $j = s$. Thus $\hat{x}'^{sk} = \hat{x}'^{s(k-1)+s}$ in (3.7) denotes value of the solution computed in finite precision at the end of the $(k-1)^{\text{st}}$ outer loop. Similarly, we use $\hat{r}'^{sk+s}$ to denote the computed value of r'^{sk+s}. At the end of outer loop k, these computed quantities can be written as

$$\begin{aligned}
\hat{x}'^{sk+s} &\equiv fl(V^k \hat{e}^{sk+s} + \hat{x}'^{sk}) = V^k \hat{e}^{sk+s} + \hat{x}'^{sk} + \psi^{sk+s} = x'^{sk+s} + \psi^{sk+s} && (3.9) \\
&\left|\psi^{sk+s}\right| \leq \epsilon \left(\left|\hat{x}'^{sk+s}\right| + N_V \left|V^k\right| \left|\hat{e}^{sk+s}\right|\right) && (3.10) \\
\hat{r}'^{sk+s} &\equiv V^k \hat{c}^{sk+s} + \phi^{sk+s} = r'^{sk+s} + \phi^{sk+s} && (3.11) \\
&\left|\phi^{sk+s}\right| \leq \epsilon N_V \left|V^k\right| \left|\hat{c}^{sk+s}\right| && (3.12)
\end{aligned}$$

Note that, similar to the above case for $\alpha_{sk+j-1} a^{sk+j-1}$, we do not account for any errors in computing the Krylov basis V^k. Again, the deviation of the computed residual, r'^{sk+s}, and the true residual, $b - Ax'^{sk+s}$, is due to the different round-off patterns that come from different treatment of V^k in the recurrences. This means that errors made in computing the basis do not contribute to the numerical deviation of the true and computed residual. A nice consequence of this is that our analysis is independent of how V^k is computed; the same bounds hold for all polynomial bases, even in the face of numerical rank deficiency (although in this case, we certainly don't expect convergence of the computed residual, which makes residual replacement efforts irrelevant).

3.2. Deviation of the true and computed residual. We now analyze round-off error that occurs in finite precision CA-CG and obtain an upper bound for the norm of the difference between the true and computed residual at step $sk+j$, denoted by δ^{sk+j}. We comment that the same analysis and bounds also apply to CA-BICG, whose true and computed residual follow the same update formulas.

In the inner loop, we perform vector updates in the s-dimensional Krylov bases represented by V^k. Using (2.3), we can write the deviation of the true residual, $b - Ax'^{sk+j}$, and the computed residual, r'^{sk+j}, as

$$b - Ax'^{sk+j} - r'^{sk+j} = b - (Ax'^{sk+j} + r'^{sk+j}) = b - V^k(T\hat{e}^{sk+j} + \hat{c}^{sk+j}) - A\hat{x}'^{sk}$$

for $1 \leq j < s$. Using (3.3) and (3.5),

$$\begin{aligned} T\hat{e}^{sk+j} + \hat{c}^{sk+j} &= T\left(\hat{e}^{sk+j-1} + \alpha_{sk+j-1}a^{sk+j-1} + \xi^{sk+j}\right) \\ &\quad + \hat{c}^{sk+j-1} - T\alpha_{sk+j-1}a^{sk+j-1} + \eta^{sk+j} \\ &= T\hat{e}^{sk+j-1} + \hat{c}^{sk+j-1} + T\xi^{sk+j} + \eta^{sk+j} \end{aligned}$$

We plug back in to obtain

$$b - V^k(T\hat{e}^{sk+j} + \hat{c}^{sk+j}) - A\hat{x}'^{sk} = b - A\hat{x}'^{sk} - \hat{r}'^{sk} - \sum_{i=1}^{j}\left(V^kT\xi^{sk+i} + V^k\eta^{sk+i}\right)$$

where we have used the base cases for the values of $\hat{e}^{sk}$ and $\hat{c}^{sk}$ given in Alg. 1.

We can then use (3.9) and (3.11) to write

$$b - A\hat{x}'^{sk} - \hat{r}'^{sk} = \left(b - Ax'^{s(k-1)+s} - r'^{s(k-1)+s}\right) - A\psi^{s(k-1)+s} - \phi^{s(k-1)+s}$$

If we keep unrolling the recurrence, we obtain an expression for the deviation of the true and computed residual:

$$b - Ax'^{sk+j} - r'^{sk+j} = b - Ax^0 - r^0 + \delta^{sk+j} \tag{3.13}$$

where

$$\begin{aligned} \delta^{sk+j} \equiv &- \sum_{l=0}^{k-1}\left[A\psi^{sl+s} + \phi^{sl+s} + \sum_{i=1}^{s}\left(V^lT\xi^{sl+i} + V^l\eta^{sl+i}\right)\right] \\ &- \sum_{i=1}^{j}\left(V^kT\xi^{sk+i} + V^k\eta^{sk+i}\right) \end{aligned} \tag{3.14}$$

We will now bound the norm of δ^{sk+j}. Using the componentwise bounds for ξ^{sl+i} and η^{sl+i} in (3.4) and (3.6), respectively, we can write

$$\begin{aligned} &\sum_{i=1}^{j}[\left|V^k\right|\left(|T|\left|\xi^{sk+i}\right| + \left|\eta^{sk+i}\right|\right)] \\ &\qquad \leq \epsilon\sum_{i=1}^{j}\left|V^k\right|\left(|T|\left|\hat{e}^{sk+i}\right| + \left|\hat{c}^{sk+i}\right| + N_T\,|T|\left|\alpha_{sk+i-1}a^{sk+i-1}\right|\right) \end{aligned}$$

Rearranging (3.3), we obtain

$$\alpha_{sk+i-1}a^{sk+i-1} = \hat{e}^{sk+i} - \hat{e}^{sk+i-1} - \xi^{sk+i}$$

which gives us the componentwise bound for the summation

$$\sum_{i=1}^{j} \left|\alpha_{sk+i-1}a^{sk+i-1}\right| \leq (2+\epsilon)\sum_{i=1}^{j} \left|\hat{e}^{sk+i}\right| \tag{3.15}$$

Therefore,

$$\sum_{i=1}^{j}[\left|V^k\right|(|T|\left|\xi^{sk+i}\right| + \left|\eta^{sk+i}\right|)] \leq \epsilon\sum_{i=1}^{j}\left|V^k\right|\left[(1+2N_T)|T|\left|\hat{e}^{sk+i}\right| + \left|\hat{c}^{sk+i}\right|\right]$$

If we combine this with the componentwise bounds for ψ^{sl+s} and ϕ^{sl+s} in (3.10) and (3.12), we get

$$\begin{aligned}\left|\delta^{sk+j}\right| \leq \epsilon\sum_{l=0}^{k-1}\left(|A|\left|\hat{x}'^{sl+s}\right| + N_V|A|\left|V^l\right|\left|\hat{e}^{sl+s}\right| + N_V\left|V^l\right|\left|\hat{c}^{sl+s}\right|\right) \\ +\epsilon\sum_{l=0}^{k-1}\sum_{i=1}^{s}\left|V^l\right|\left[(1+2N_T)|T|\left|\hat{e}^{sl+i}\right| + \left|\hat{c}^{sl+i}\right|\right] \\ +\epsilon\sum_{i=1}^{j}\left|V^k\right|\left[(1+2N_T)|T|\left|\hat{e}^{sk+i}\right| + \left|\hat{c}^{sk+i}\right|\right]\end{aligned} \tag{3.16}$$

From this follows a bound for δ^{sk+j} in terms of norms:

$$\begin{aligned}\left\|\delta^{sk+j}\right\| \leq \epsilon\sum_{l=0}^{k-1}\left(\|A\|\left\|\hat{x}'^{sl+s}\right\| + N_V\|A\|\left\|\left|V^l\right|\left|\hat{e}^{sl+s}\right|\right\| + N_V\left\|\left|V^l\right|\left|\hat{c}^{sl+s}\right|\right\|\right) \\ +\epsilon\sum_{l=0}^{k-1}\sum_{i=1}^{s}\left[(1+2N_T)\|T\|\left\|\left|V^l\right|\left|\hat{e}^{sl+i}\right|\right\| + \left\|\left|V^l\right|\left|\hat{c}^{sl+i}\right|\right\|\right] \\ +\epsilon\sum_{i=1}^{j}\left[(1+2N_T)\|T\|\left\|\left|V^k\right|\left|\hat{e}^{sk+i}\right|\right\| + \left\|\left|V^k\right|\left|\hat{c}^{sk+i}\right|\right\|\right]\end{aligned} \tag{3.17}$$

In the equation above, it could be the case that $\left\|V^k\right\|\left\|\hat{e}^{sk+j}\right\| \gg \left\|\left|V^k\right|\left|\hat{e}^{sk+j}\right|\right\|$ and $\left\|V^k\right\|\left\|\hat{c}^{sk+j}\right\| \gg \left\|\left|V^k\right|\left|\hat{c}^{sk+j}\right|\right\|$, which will lead to a very large overestimate of the deviation of residuals. Therefore it is beneficial to use the quantities $\left\|\left|V^k\right|\left|\hat{e}^{sk+j}\right|\right\|$ and $\left\|\left|V^k\right|\left|\hat{c}^{sk+j}\right|\right\|$ in (3.17).

4. Error in finite precision CA-KSMs with residual replacement and group update. In the residual replacement strategy of Van der Vorst and Ye [24], the computed residual $\hat{r}^m$ is replaced with the true residual $fl(b - A\hat{x}^m)$ at *residual replacement* steps $m = m_1, m_2, \cdots, m_y$.

Van der Vorst and Ye combine residual replacement with a *group update* strategy. A group update strategy ensures that the deviation of residuals remains on the order of $O(\epsilon)\|A\|\|x\|$ from the last replacement step m_k to the final iteration by reducing

error in the local recurrence. Similar techniques are used to reduce error accumulation in evaluating a sum $S = \sum_{i=1}^{\infty} w_i$ of small numbers by direct recursive additions [24]. This error can be corrected by grouping operations as $S_1 + S_2 + \cdots = (w_1 + w_2 + \cdots + w_{m_1}) + (w_{m_1+1} + \cdots + w_{m_2}) + \ldots$, where terms close in magnitude are grouped in S_i. This makes rounding error associated with additions within group S_i of magnitude ϵS_i, which can be much smaller than ϵS. This strategy has previously been suggested for use in KSMs by Neumaier and was also used by Sleijpen and Van der Vorst (see [17, 20, 24]), where the solution is instead updated by the recurrence

$$\begin{aligned} x^n &= x^0 + \sum_{i=1}^{n} \alpha_{i-1} a^{i-1} \\ &= x^0 + \left(\alpha_0 a^0 + \cdots + \alpha_{m_1-1} a^{m_1-1}\right) + \left(\alpha_{m_1} a^{m_1} + \cdots + \alpha_{m_2-1} a^{m_2-1}\right) + \ldots \end{aligned}$$

In [24], when a residual replacement occurs, a group update is also performed; we use the same strategy here. When a residual replacement and group update occurs in CA-CG, we update vector z, which represents the group approximate solution, as $z = z + x'^{sk+j}$, and reset $x'^{sk+j} = x^0 = 0$. The true residual is then computed as $fl(b - Az)$.

In the CA-CG algorithm, we begin a new outer loop immediately following a residual replacement step, as we can no longer use the generated s-step bases to recover iterates according to the recurrence; we must redistribute the true residual and generate a new basis using the matrix powers kernel. For simplicity of notation, we will reset $k = 0$ after a group update occurs. Although this adds additional communication steps to the algorithm, it will be shown in subsequent sections that the number of replacements is small, and thus the extra communication cost is negligible.

In order for our residual replacement strategy to be successful, we must satisfy the following constraints [24]:

1. The accumulated error from the last residual replacement step m_y should be small relative to $\epsilon(|r'^{m_y}| + N_A\,|A|\,|x'^{m_y}|)$.
2. The residual in the algorithm with residual replacement should maintain the convergence mechanism driven by underlying finite precision Lanczos process.

The use of the group update strategy satisfies the first objective [24]. We briefly discuss the latter constraint, which will motivate the condition for residual replacement.

4.1. Selecting residual replacement steps. We review the analysis in [24] which leads to the residual replacement condition. Consider the classical finite precision CG iteration, where, in iteration n, the computed residuals $\hat{r}^n$ and search directions $\hat{p}^n$ satisfy

$$\hat{r}^n = \hat{r}^{n-1} - \alpha_{n-1} A\hat{p}^{n-1} + \eta^n \qquad \hat{p}^n = \hat{r}^n + \beta_n \hat{p}^{n-1} + \tau^n \tag{4.1}$$

In this subsection, we let e_n denote the n^{th} identity column, rather than the coefficients used in CA-CG. We can then write the above equations in matrix form as

$$AZ_n = Z_n H_n - \frac{1}{\alpha'_n} \frac{\hat{r}^{n+1}}{\|\hat{r}^1\|} e_n^T + F_n \quad \text{with} \quad Z_n = \left[\frac{\hat{r}^1}{\|\hat{r}^1\|}, \ldots, \frac{\hat{r}^n}{\|\hat{r}^n\|}\right]$$

where H_n is an invertible tridiagonal matrix, $\alpha'_n = e_n^T H_n^{-1} e_1$ and $F_n = \left[f^1, \ldots, f^n\right]$ with

$$f^n = \frac{A\tau^n}{\|\hat{r}^n\|} + \frac{1}{\alpha_n} \frac{\eta^{n+1}}{\|\hat{r}^n\|} - \frac{\beta_n}{\alpha_{n-1}} \frac{\eta^n}{\|\hat{r}^n\|} \tag{4.2}$$

It has been shown (see, e.g., [14]) that if a sequence $\hat{r}^n$ satisfies (4.1) and the basis Z^{n+1} is full rank,

$$\left\|\hat{r}^{n+1}\right\| \leq (1+K_n) \min_{\rho\in\mathcal{P}_n,\rho(0)=1} \left\|\rho(A+\Delta A_n)\hat{r}^1\right\| \tag{4.3}$$

where $\mathcal{P}_n$ is the set of polynomials of degree n, $K_n = \left\|(AZ_n - F_n)H_n^{-1}\right\| \left\|Z_{n+1}^+\right\|$ and $\Delta A_n = -F_n Z_n^+$. A consequence of this is that no matter how $\hat{r}^n$ is generated, if it satisfies (4.1), we can bound its norm by (4.3). Then by (4.2), we can replace the computed residual with the true residual without affecting the convergence rate when η^n is not too large relative to $\|\hat{r}^n\|$ and $\left\|\hat{r}^{n-1}\right\|$.

We now seek a bound on the perturbation term in CA-CG with residual replacement and group update strategies, which we denote $\eta^n_{(\mathrm{RR})}$, analogous to η^n term for the algorithm without replacement. Comparing this value with the norms of the computed residuals in the CA-CG algorithm with replacement, $\|\hat{r}^n\|$ and $\left\|\hat{r}^{n-1}\right\|$, will then allow us to select safe residual replacement steps. Specifically, this gives the following condition for residual replacement:

$$\left\|\eta^{n-1}_{(\mathrm{RR})}\right\| \leq \tau \left\|\hat{r}^{n-1}\right\| \textbf{ and } \left\|\eta^{n}_{(\mathrm{RR})}\right\| > \tau \left\|\hat{r}^{n}\right\| \tag{4.4}$$

where τ is a tolerance parameter. Because τ controls perturbations to the Lanczos recurrence, it should be chosen as small as possible. However, if it is too small, residual replacement will terminate early in the iteration and the accumulated error after the last replacement can become significant [24]. The value $\tau = \sqrt{\epsilon}$ has been found to balance these two constraints for standard KSMs [24]; we have observed good results for CA-KSMs as well.

4.2. Bounding the error term. We will now describe and analyze the error in CA-CG with residual replacement and group update strategies. Our goal is a bound on the perturbation term $\eta^{sk+j}_{(\mathrm{RR})}$ in the recurrence for the computed residual in the CA-KSM with residual replacement and group update:

$$\hat{r}^n = V^k\left(\hat{c}^{sk+j-1} - T\alpha_{sk+j-1}a^{sk+j-1}\right) + \eta^{sk+j}_{(\mathrm{RR})}$$

This will enable us to determine when we can replace the computed residual with the true residual without affecting convergence. We note that the same analysis applies to CA-BICG. We use r^n to denote the residual in the CA-KSM with residual replacement and group updating, defined as

$$r^n = \begin{cases} fl(b - Ax^n), & n = m_1, \cdots, m_y \\ r'^n = V^k\left(\hat{c}^{sk+j-1} - T\left(\alpha_{sk+j-1}a^{sk+j-1}\right) + \eta^{sk+j}\right), & \text{otherwise} \end{cases} \tag{4.5}$$

At iteration $n = sk + j$, where m was the last residual replacement step, we use x^n to denote the current solution in the residual replacement algorithm, that is,

$$x^n = x^m + x'^{sk+j} = x^m + \hat{x}'^{sk} + V^k\hat{e}^{sk+j} \tag{4.6}$$

where x^m is the approximate solution computed in the last group update in step m and x'^{sk+j} remains defined as in (3.7).

We use $\hat{r}^n$ to denote the finite precision evaluation of the computed residual in (4.5) and $\hat{x}^n$ to denote the finite precision evaluation of the approximate solution in (4.6). If we replace the residual and perform a group update in iteration n, we can write the computed group approximate solution and residual as

$$\hat{x}^n \equiv fl(x^m + x'^{sk+j}) = x^m + x'^{sk+j} + \zeta^n \tag{4.7}$$

$$\text{where } |\zeta^n| \leq \epsilon\,|x^n| \leq \epsilon\left(|x^m| + \left|\hat{x}'^{sk}\right| + \left|V^k \hat{e}^{sk+j}\right|\right) \tag{4.8}$$

$$\begin{aligned}\hat{r}^n &\equiv fl(b - A\hat{x}^n) = b - A\left(x^m + \hat{x}'^{sk} + V^k\hat{e}^{sk+j} + \zeta^n\right) + \mu^n \\ &= b - Ax^n - A\zeta^n + \mu^n \end{aligned} \tag{4.9}$$

$$\begin{aligned}\text{where } |\mu^n| &\leq \epsilon\left(|r^n| + N_A\,|A|\,|x^n|\right) \\ &\leq \epsilon\left(|r^n| + N_A\,|A|\,|x^m| + N_A\,|A|\left|\hat{x}'^{sk}\right| + N_A\,|A|\left|V^k\hat{e}^{sk+j}\right|\right)\end{aligned} \tag{4.10}$$

Immediately after the last residual replacement step m, $x^m = z + x^0 = z$, where z is the updated solution vector and x^0 has been set to 0 (note that $\hat{e}^0 = 0_{2s+1}$, since we begin a new outer loop after replacement). Then at step m,

$$r^m = fl(b - Az) = fl(b - Ax^m) = b - Ax^m + \mu^m$$

Let $\delta^m_{(\mathrm{RR})} = b - Ax^m - r^m$ denote the deviation of the true and computed residual in CA-CG with residual replacement. Then we can rearrange the above to obtain $\delta^m_{(\mathrm{RR})} = b - Ax^m - r^m = -\mu^m$. Because we computed $r^m = fl(b - Ax^m)$ by (4.5), we can bound μ^m as $|\mu^m| \leq \epsilon\,(|r^m| + N_A\,|x^m|)$. Then, after a residual replacement step, the norm of the deviation of residuals is immediately reduced to $\|\mu^m\|$.

Now, at step $n = sk + j$, where m was the last residual replacement step, we can write the deviation of the true and computed residuals as

$$\begin{aligned}\delta^n_{(\mathrm{RR})} &= b - Ax^n - r^n = b - A\left(x^m + x'^{sk+j}\right) - r^n \\ &= b - Ax^m - A\hat{x}'^{sk+j} - \hat{r}'^{sk+j} \\ &= -\mu^m + \delta^{sk+j}\end{aligned} \tag{4.11}$$

where we have used the results in (3.14). Using the above, we obtain

$$\begin{aligned}b - Ax^n &= r^n + \delta^n_{(\mathrm{RR})} = V^k\left(\hat{c}^{sk+j-1} - T\alpha_{sk+j-1}a^{sk+j-1} + \eta^n\right) + \delta^n_{(\mathrm{RR})} \\ &= r^{n-1} + V^k\eta^n + \delta^n_{(\mathrm{RR})}\end{aligned} \tag{4.12}$$

If we take this expression for $b - Ax^n$ and substitute into (4.9), we obtain a recurrence for the residual in finite precision CA-CG with residual replacement and group update strategies:

$$\hat{r}^n = V^k\left(\hat{c}^{sk+j-1} - T\alpha_{sk+j-1}a^{sk+j-1}\right) + \eta^n_{(\mathrm{RR})} \tag{4.13}$$

where

$$\eta^{sk+j}{}_{(\mathrm{RR})} \equiv \eta^n{}_{(\mathrm{RR})} = V^k\eta^n + \delta^n_{(\mathrm{RR})} - A\zeta^n + \mu^n \tag{4.14}$$

Substituting in the expression for $\delta^n_{(\mathrm{RR})}$ in (4.11), we can write $\delta^n_{(\mathrm{RR})} + V^k\eta^n = -\mu^m + \delta^{sk+j} + V^k\eta^{sk+j}$. Then, using the analysis in the previous section, namely (3.16) and (3.17), we can bound the norm of this quantity as

$$\begin{aligned}\left\|\delta^n_{(\mathrm{RR})} + V^k\eta^n\right\| \leq \|\mu^m\| &\\ +\epsilon\sum_{l=0}^{k-1}\left(\|A\|\left\|\hat{x}'^{sl+s}\right\| + N_V\|A\|\left\|\left|V^l\right|\left|\hat{e}^{sl+s}\right|\right\| + N_V\left\|\left|V^l\right|\left|\hat{c}^{sl+s}\right|\right\|\right)&\\ +\epsilon\sum_{l=0}^{k-1}\sum_{i=1}^{s}\left[(1+2N_T)\|T\|\left\|\left|V^l\right|\left|\hat{e}^{sl+i}\right|\right\| + \left\|\left|V^l\right|\left|\hat{c}^{sl+i}\right|\right\|\right]&\\ +\epsilon\sum_{i=1}^{j}\|T\|\left\|\left|V^k\right|\left|\hat{e}^{sk+i}\right|\right\|&\\ +\epsilon\sum_{i=1}^{j-1}\left[2N_T\|T\|\left\|\left|V^k\right|\left|\hat{e}^{sk+i}\right|\right\| + \left\|\left|V^k\right|\left|\hat{c}^{sk+i}\right|\right\|\right] &\quad (4.15)\end{aligned}$$

Using (4.10), we can bound $\|\mu^n\|$ by

$$\begin{aligned}\|\mu^n\| &\leq \epsilon\left\||r^n| + N_A|A||x^m| + N_A|A|\left|\hat{x}'^{sk}\right| + N_A|A|\left|V^k\hat{e}^{sk+j}\right|\right\| \\ &\leq \epsilon\left(\|r^n\| + N_A\|A\|\|x^m\| + N_A\|A\|\left\|\hat{x}'^{sk}\right\| + N_A\left\||A|\left|V^k\hat{e}^{sk+j}\right|\right\|\right) \quad (4.16)\end{aligned}$$

and using (4.8), we can bound $\|A\zeta^n\|$ by

$$\begin{aligned}\|A\zeta^n\| &\leq \left\||A||x^m| + |A|\left|x'^{sk}\right| + |A|\left|V^k\hat{e}^{sk+j}\right|\right\| \\ &\leq \epsilon\left(\|A\|\|x^m\| + \|A\|\left\|\hat{x}'^{sk}\right\| + \left\||A|\left|V^k\hat{e}^{sk+j}\right|\right\|\right) \quad (4.17)\end{aligned}$$

Because we can approximate quantities $\|r^n\| \leq \left\|\left|V^k\right|\left|\hat{c}^{sk+j}\right|\right\|$ and

$$(1+N_A)\left\||A|\left|V^k\hat{e}^{sk+j}\right|\right\| \leq 2N_T\|T\|\left\|\left|V^k\right|\left|\hat{e}^{sk+j}\right|\right\|$$

the above can be simplified to

$$\begin{aligned}\left\|\eta^n_{(\mathrm{RR})}\right\| &\leq \left\|\delta^n_{(\mathrm{RR})} + V^k\eta^n\right\| + \|A\zeta^n\| + \|\mu^n\| \\ &\leq \epsilon\left(\|r^m\| + (1+2N_A)\|A\|\|x^m\|\right) \\ &\quad +\epsilon\sum_{l=0}^{k-1}\left((2+N_A)\|A\|\left\|\hat{x}'^{sl+s}\right\| + N_V\|A\|\left\|\left|V^l\right|\left|\hat{e}^{sl+s}\right|\right\| + N_V\left\|\left|V^l\right|\left|\hat{c}^{sl+s}\right|\right\|\right) \\ &\quad +\epsilon\sum_{l=0}^{k-1}\sum_{i=1}^{s}\left[(1+2N_T)\|T\|\left\|\left|V^l\right|\left|\hat{e}^{sl+i}\right|\right\| + \left\|\left|V^l\right|\left|\hat{c}^{sl+i}\right|\right\|\right] \\ &\quad +\epsilon\sum_{i=1}^{j}\left[(1+2N_T)\|T\|\left\|\left|V^k\right|\left|\hat{e}^{sk+i}\right|\right\| + \left\|\left|V^k\right|\left|\hat{c}^{sk+i}\right|\right\|\right] \\ &\equiv d_{sk+j} \quad (4.18)\end{aligned}$$

where d_{sk+j} is a scalar that will upper bound the perturbation term $\eta^n_{(\mathrm{RR})}$ in our algorithm. Again, it is beneficial to compute the componentwise product before taking norms when $\left\|V^k\right\|$ is very large. Therefore we use the quantities $\left\|\left|V^k\right|\left|\hat{e}^{sk+j}\right|\right\|$ and $\left\|\left|V^k\right|\left|\hat{c}^{sk+j}\right|\right\|$ in (4.18). The additional communication and computation costs of computing these tighter bounds is discussed in the following section.

4.3. Avoiding communication in computing the upper bound. In each iteration, we will update d_{sk+j}, the deviation of the true and computed residual, given by (4.18). We show that d_{sk+j} can be estimated in each iteration in a communication-avoiding way for little extra computational cost. Notice in (4.18) that d_{sk+j} is computable in each iteration with quantities already available (either local to each processor or in fast memory). We can iteratively update d_{sk+j} according to

$$d_{sk+j} = d_{sk+j-1} + \epsilon \left[(1+2N_T) \|T\| \left\| \left| V^k \right| \left| \hat{e}^{sk+j} \right| \right\| + \left\| \left| V^k \right| \left| \hat{c}^{sk+j} \right| \right\| \right] + \epsilon \begin{cases} (2+N_A) \|A\| \left\| \hat{x}'^{sk+s} \right\| + N_V \|A\| \left\| \left| V^k \right| \left| \hat{e}^{sk+s} \right| \right\| \\ \quad + N_V \left\| \left| V^k \right| \left| \hat{c}^{sk+s} \right| \right\| & j = s \\ 0 & \text{otherwise} \end{cases} \tag{4.19}$$

where we set $d_0 = \epsilon \left(\|r^m\| + (1+2N_A) \|A\| \|x^m\| \right)$ when residual replacement occurs at iteration m.

We therefore need estimates for the quantities $\left\| \left| V^k \right| \left| \hat{e}^{sk+j} \right| \right\|$, $\left\| \left| V^k \right| \left| \hat{c}^{sk+j} \right| \right\|$, $\|T\|$, $\|A\|$, $\|r^m\|$, $\|x^m\|$, and $\left\| \hat{x}'^{sk+s} \right\|$ in order to compute d_{sk+j}. We discuss how to estimate each of these quantities without increasing the asymptotic computation or communication cost of the CA-KSM iterations.

We can assume that we have estimates for $\|A\|$ and $\|T\|$. These need only be computed once, since A and T remain the same throughout the iterations (note that if we were to iteratively update our basis parameters every s steps, T would change between outer iterations. In this case, we can still approximate $\|T\|$ cheaply, since it is small and tridiagonal.

To compute $\left\| \left| V^k \right| \left| \hat{e}^{sk+j} \right| \right\|$ and $\left\| \left| V^k \right| \left| \hat{c}^{sk+j} \right| \right\|$ locally, we need to define a new quantity, $\tilde{G}^k = \left| V^k \right|^T \left| V^k \right|$. Computing $\tilde{G}^k$ requires an all-to-all reduction, which adds $O(ns^2)$ operations and requires moving $O(s^2)$ words per outer loop. Because these are the same costs as computing G^k in each outer loop, using $\tilde{G}^k$ does not affect the asymptotic cost of communication or computation. Using this quantity,

$$\left\| \left| V^k \right| \left| \hat{e}^{sk+j} \right| \right\|_2 = \sqrt{\left| \hat{e}^{sk+j} \right|^T \tilde{G}^k \left| \hat{e}^{sk+j} \right|} \qquad \left\| \left| V^k \right| \left| \hat{c}^{sk+j} \right| \right\|_2 = \sqrt{\left| \hat{c}^{sk+j} \right|^T \tilde{G}^k \left| \hat{c}^{sk+j} \right|}$$

which can be computed locally in $O(s^3)$ operations per s steps. Note that, until now, we have not assumed a specific norm for the bound. Although (4.18) holds generally for any p-norm, in practice we use the 2-norm, since this allows the use of shortened dot products between basis vectors, and thus maintains the communication-avoiding properties of the original CA-KSM.

When we replace the residual, we must communicate the computed values of x^m and r^m between processors, so we can compute $\|x^m\|$ and $\|r^m\|$ at the same time for $O(n)$ total operations per outer loop. The same applies for $\left\| \hat{x}'^{sk+s} \right\|$ since $\hat{x}'^{sk+s}$ is already computed and communicated between processors at the end of each inner loop when $j = s$.

Accounting for all the terms above, the total cost of computing d_{sk+j} in each iteration is $O(ns^2)$ flops and requires moving $O(s^2+n)$ words per s iterations, assuming $s \ll n$. In the CA-KSM without residual replacement, the computation cost is $\Omega(ns^2)$ and the communication cost is $\Omega(s^2+n)$. The lower bounds here represent the cost of the dense work. The communication and computation performed by the matrix powers kernel is dependent on the structure of A. From this, we see that if we combine the computation of d_{sk+j} with the CA-KSM as in our residual replacement strategy, the asymptotic communication and computation costs of s steps of the algorithm are

the same as for the CA-KSM alone. Therefore, we conclude that we can update the quantity d_{sk+j} in each iteration efficiently.

4.4. Residual replacement algorithm for CA-KSMs. Based on (4.4), Van der Vorst and Ye [24] use the following residual replacement condition in their implementation:

$$d_{sk+j-1} \leq \tau \left\|\hat{r}^{sk+j-1}\right\| \text{ and } d_{sk+j} > \tau \left\|\hat{r}^{sk+j}\right\| \text{ and } d_{sk+j} > 1.1 d_{\text{init}} \tag{4.20}$$

where they initially set $d_0 = d_{\text{init}} = \epsilon(\left\|r^0\right\| + N_A \|A\| \left\|x^0\right\|) = \epsilon \|b\|$, and reset $d_{\text{init}} = \epsilon(\|r^m\| + N_A \|A\| \|x^m\|$ when replacement occurs. The third added constraint avoids unnecessary replacements and ensures there has been a nontrivial increase in error since the last replacement step [24].

In our case, in the inner loop, we do not know the actual values of $\hat{r}^{sk+j-1}$ and $\hat{r}^{sk+j}$ as they are not computed, but we can use the Gram matrix G^k to estimate their 2-norms as $\left\|\hat{r}^{sk+j-1}\right\|_2 \approx \left\|V^k \hat{c}^{sk+j-1}\right\|_2 \approx \sqrt{(\hat{c}^{sk+j-1})^T G^k (\hat{c}^{sk+j-1})}$ and $\left\|\hat{r}^{sk+j}\right\|_2 \approx \left\|V^k \hat{c}^{sk+j}\right\|_2 \approx \sqrt{(\hat{c}^{sk+j})^T G^k (\hat{c}^{sk+j})}$. This again suggests the use of the 2-norm in efficient implementation of the residual replacement algorithm. We note that using the Gram matrix to compute inner products locally does result in an extra $O(\epsilon)$ error term. However, since we will multiply this term by $\tau = \epsilon^{1/2}$ in the residual replacement condition, we can ignore this higher power of ϵ.

Our condition for residual replacement in CA-KSMs will then be

$$d_{sk+j-1} \leq \tau \left\|V^k \hat{c}^{sk+j-1}\right\| \text{ and } d_{sk+j} > \tau \left\|V^k \hat{c}^{sk+j}\right\| \text{ and } d_{sk+j} > 1.1 d_{\text{init}} \tag{4.21}$$

where we use the same initial condition as in [24] and update $d_{\text{init}} = \epsilon(\|r^m\| + N_A \|A\| \|x^m\|$ at replacement step m by (4.18). If this statement is true, we accumulate the current value of x'^{sk+j} into vector z, as $z = fl(z + x'^{sk+j})$, and we set $r^{sk+j} = \hat{r}^{sk+j} = fl(b - Az)$.

To perform a residual replacement in CA-KSMs, all processors must communicate their elements of x'^{sk+j} to compute $b - Az$, and we must break out of the inner loop (potentially before completing s steps) and continue with computing the next matrix powers kernel with the new residual in the next outer loop. This means our communication costs could potentially increase if the number of replacements is high (i.e., we compute the true residual every iteration), but our experimental results in the next section indicate that the number of replacements is low compared to the total number of iterations. Therefore the communication cost does not asymptotically increase versus the CA-KSM without residual replacement.

We can now write CA-CG with residual replacement, shown in Alg. 2.

5. Numerical experiments. We evaluated our residual replacement strategy on a number of matrices from the University of Florida Sparse Matrix Collection [6]. We present a subset of our experiments here which demonstrate the general trends observed. The selected matrices, listed along with their properties in Table 5.1, are representative of a variety of problem domains, including thermal problems, structural problems, finite element models, acoustics problems, and circuit simulations. We tested both symmetric and unsymmetric matrices, using the CA-CG and CA-BICG methods, respectively.

In the experiments, we compare classical (BI)CG with CA-(BI)CG, both with and without residual replacement. For the CA-KSMs, tests were run for $s = [4, 8, 12]$, with the monomial, Newton and Chebyshev bases. We used row and column scaling

Algorithm 2 CA-CG with Residual Replacement

Compute $\|A\|$
$x^0 = 0$, $r^0 = b - Ax^0$, $p^0 = r^0$, $z = 0$, $k = 0$, reset $= 0$
$d_0 = d_{\text{init}} = \epsilon(\|r^0\| + N_A \|A\| \|x^0\|)$
while not converged **do**
  Calculate P^k, R^k, T by (2.1) and (2.3)
  if $k == 0$ **then** compute $\|T\|$ **end if**
  Let $V^k = [P^k, R^k]$, $G^k = (V^k)^T V^k$, $\tilde{G}^k = |V^k|^T |V^k|$
  $a^{sk} = [1, 0_{2s}]^T$, $c^{sk} = [0_{s+1}, 1, 0_{s-1}]^T$, $e^{sk} = [0_{2s+1}]$
  for $j = 1 : s$ **do**
    $\alpha_{sk+j-1} = [(c^{sk+j-1})^T G^k (c^{sk+j-1})]/[(a^{sk+j-1})^T G^k (T a^{sk+j-1})]$
    $e^{sk+j} = e^{sk+j-1} + \alpha_{sk+j-1} a^{sk+j-1}$
    $c^{sk+j} = c^{sk+j-1} - \alpha_{sk+j-1} T a^{sk+j-1}$
    $\beta_{sk+j-1} = ((c^{sk+j})^T G^k (c^{sk+j}))/((c^{sk+j-1})^T G^k (c^{sk+j-1}))$
    $a^{sk+j} = c^{sk+j} + \beta_{sk+j-1} a^{sk+j-1}$
    Update d_{sk+j} by (4.19)
    if condition (4.21) holds **then**
      $z = z + x^{sk+j}$, $r^{sk+j} = b - Az$, $x^{sk+j} = 0$
      $d_{\text{init}} = d_{sk+j} = \epsilon(\|r^{sk+j}\| + (1 + 2N_A) \|A\| \|z\|)$
      reset $= 1$, **break**
    end if
  end for
  if reset $!= 1$ **then**
    Update d_{sk+s} by (4.19)
    $x^{sk+s} = [P^k, R^k] e^{sk+s} + x^{sk}$, $r^{sk+s} = [P^k, R^k] c^{sk+s}$
  end if
  $p^{sk+s} = [P^k, R^k] a^{sk+s}$, $k = k + 1$, reset $= 0$
end while
return $z + x^{sk}$

of the input matrix A as an equilibration routine, preserving symmetry for the SPD case, as described in [11]. For each matrix, we selected a right hand side b such that $\|x\|_2 = 1$, $x_i = 1/\sqrt{N}$. All experiments were performed in double precision.

Van der Vorst and Ye [24] use $\tau = 10^{-8} \approx \sqrt{\epsilon}$ and $N_A = 1$ in their experiments (under the assumption that A is very sparse). We use $\tau = 10^{-8}$ for most experiments; for tests using the monomial basis with $s = 12$, using $\tau = 10^{-8}$ causes restarts to occur more frequently or sooner than necessary, likely due to an overestimate of the quantity $\eta^{sk+j}_{(\text{RR})}$ by d_{sk+j} as defined in (4.18). To balance this overestimate, we use the slightly lower value $\tau = s \cdot 10^{-8}$, which achieves the desired effect. As in [24], we set $N_A = 1$. We also set $N_V = N_T = 1$, which are adequate approximations since s is small.

Note that our bounds hold regardless of the polynomial basis or algorithm used in constructing V^k, so our algorithm requires no modification for use of the Newton and Chebyshev bases. The quality of the computed basis does, however, affect the convergence rate of the CA-KSM, as our experiments show. In the extreme case of a degenerate basis, the Lanczos process break downs, causing divergence of the computed residual. In this case, we do not expect to benefit from maintaining agreement between the true and computed residual, as the computed residual does not converge.

Matrix	Domain	N	nnz	cond	2-norm	SPD?
bundle	graphics/3D vision	$1.1 \cdot 10^4$	$7.7 \cdot 10^5$	$4.1 \cdot 10^1$	2.9	Y
consph	2D/3D problem	$8.3 \cdot 10^4$	$6.0 \cdot 10^6$	$9.7 \cdot 10^3$	9.7	Y
FEM	3D FEM	$1.5 \cdot 10^5$	$3.5 \cdot 10^6$	$2.1 \cdot 10^1$	3.5	N
circuit	circuit simulation	$1.5 \cdot 10^5$	$7.3 \cdot 10^5$	$2.3 \cdot 10^5$	2.0	Y
ship	structural	$1.4 \cdot 10^5$	$3.6 \cdot 10^6$	$3.2 \cdot 10^5$	4.9	Y
thermal	thermal	$8.3 \cdot 10^4$	$5.7 \cdot 10^5$	$3.0 \cdot 10^5$	1.9	Y
xenon	materials	$4.9 \cdot 10^4$	$1.2 \cdot 10^6$	$3.3 \cdot 10^4$	3.2	N

TABLE 5.1

Matrices used in test cases. From the University of Florida Sparse Matrix Collection [6]. Norm and condition numbers reflect the equilibrated system.

Table 5.2 shows the number of replacement iterations which occurred, as well as the total number of iterations for each test. Table 5.3 gives the total number of communication steps (a multiple of the latency cost) that were needed in each residual replacement experiment to converge to $O(\epsilon) \|A\| \|x\|$, along with the communication step ratio, or the number of communication steps in the classical method divided by the CA variant, both with residual replacement. The 2-norm was used for calculating all quantities in d_{sk+j}, as described in Sec. 4.3.

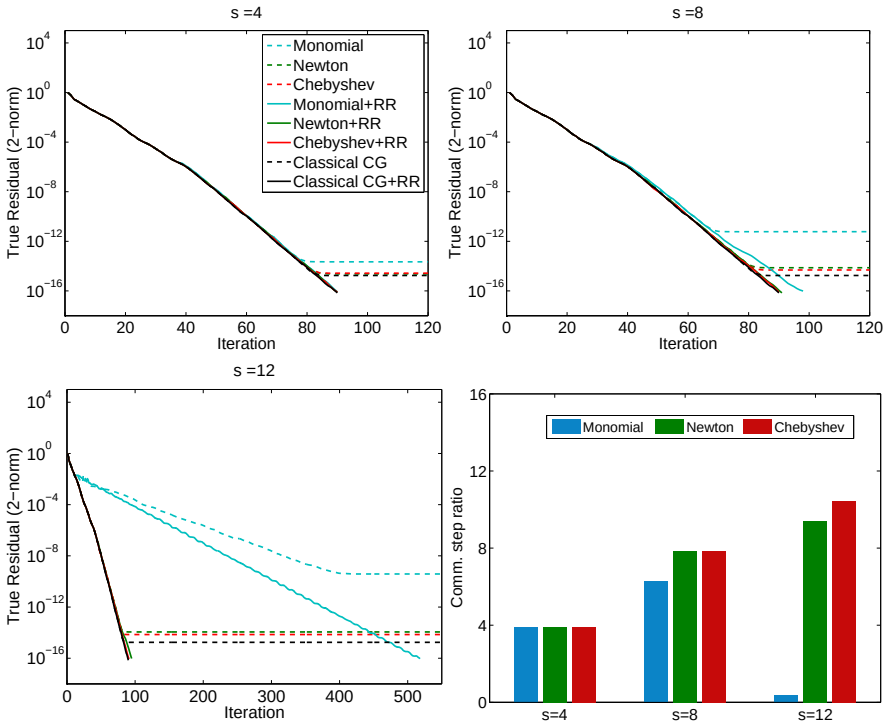

FIG. 5.1. *Convergence of true residual, bundle. Note the x-axis for $s = 12$ differs from $s = [4, 8]$.*

	Classical		Monomial						Newton						Chebyshev					
			$s=4$		$s=8$		$s=12$		$s=4$		$s=8$		$s=12$		$s=4$		$s=8$		$s=12$	
bundle	1	93	1	93	2	110	4	2803	1	95	1	97	2	104	1	95	1	96	1	104
consph	2	2215	2	2226	5	2337	5	11399	2	2227	2	2247	3	2356	2	2227	2	2235	3	2349
FEM	1	91	1	91	2	97	-	-	1	92	1	98	1	97	1	91	1	97	1	96
circuit	2	2679	2	2915	6	3773	6	62758	2	2914	2	2700	2	2918	2	2915	2	2919	2	2921
ship	3	6964	3	6997	10	8005	10	271210	3	7020	3	7413	4	7531	3	7010	3	7040	4	7512
thermal	2	2606	2	2164	7	2876	-	-	2	2168	2	2291	2	2369	2	2164	2	2166	2	2172
xenon	2	2209	2	2207	5	2433	5	3821	2	2209	2	2213	2	2213	2	2215	2	2211	2	2213

TABLE 5.2

Number of Residual Replacement Steps for CA-(BI)CG Experiments. For each test, the left column gives the total number of replacement steps and the right column shows the total number of iterations.

	Classical	Monomial						Newton						Chebyshev					
		$s=4$		$s=8$		$s=12$		$s=4$		$s=8$		$s=12$		$s=4$		$s=8$		$s=12$	
bundle	94	24	3.92	15	6.27	235	0.40	24	3.92	12	7.83	10	9.40	24	3.92	12	7.83	9	10.44
consph	2217	557	3.98	294	7.54	953	2.33	557	3.98	282	7.86	198	11.20	558	3.97	280	7.92	197	11.25
FEM	92	23	4.00	13	7.08	-	-	23	4.00	13	7.07	8	11.50	22	4.18	12	7.67	8	11.50
circuit	2681	729	3.68	474	5.66	5233	0.51	728	3.68	338	7.93	244	10.99	729	3.68	366	7.33	244	10.99
ship	6967	1750	3.98	1005	6.93	22605	0.31	1757	3.97	928	7.51	630	11.06	1754	3.97	881	7.91	627	11.11
thermal	2068	542	3.82	363	5.70	-	-	543	3.81	288	7.18	198	10.44	541	3.82	271	7.63	182	11.36
xenon	2211	553	4.00	306	7.23	321	6.89	553	4.00	277	7.98	185	11.95	555	3.98	277	7.98	186	11.89

TABLE 5.3

Total number of communication steps. For each test, the left column gives the total number of communication steps and the right column shows the total communication steps in the classical algorithm divided by the total communication steps for the CA algorithm with given s value. This value should ideally be s.

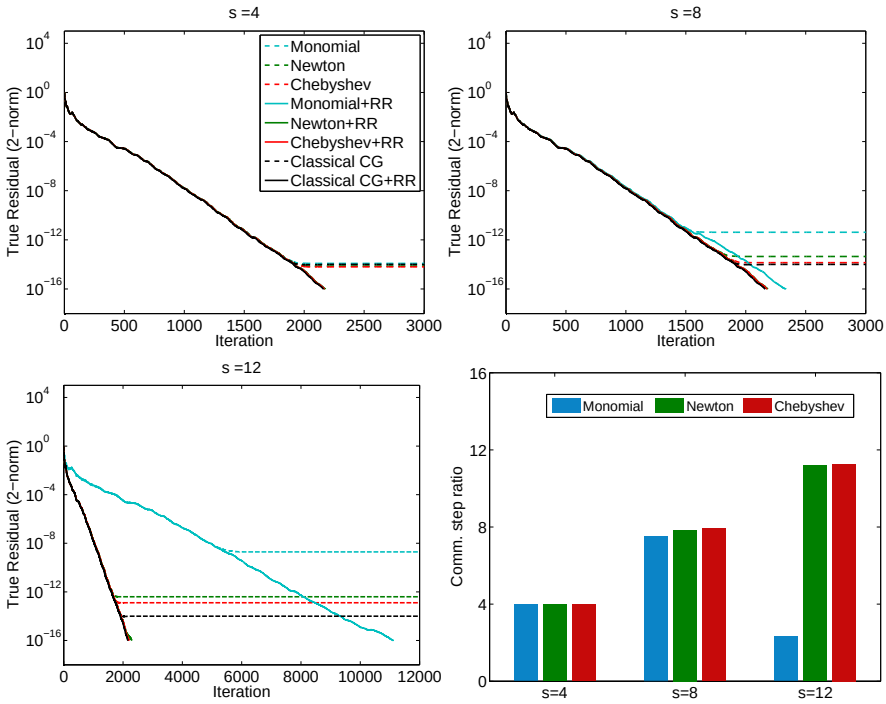

FIG. 5.2. *Convergence of true residual, consph. Note the x-axis for* $s = 12$ *differs from* $s = [4, 8]$.

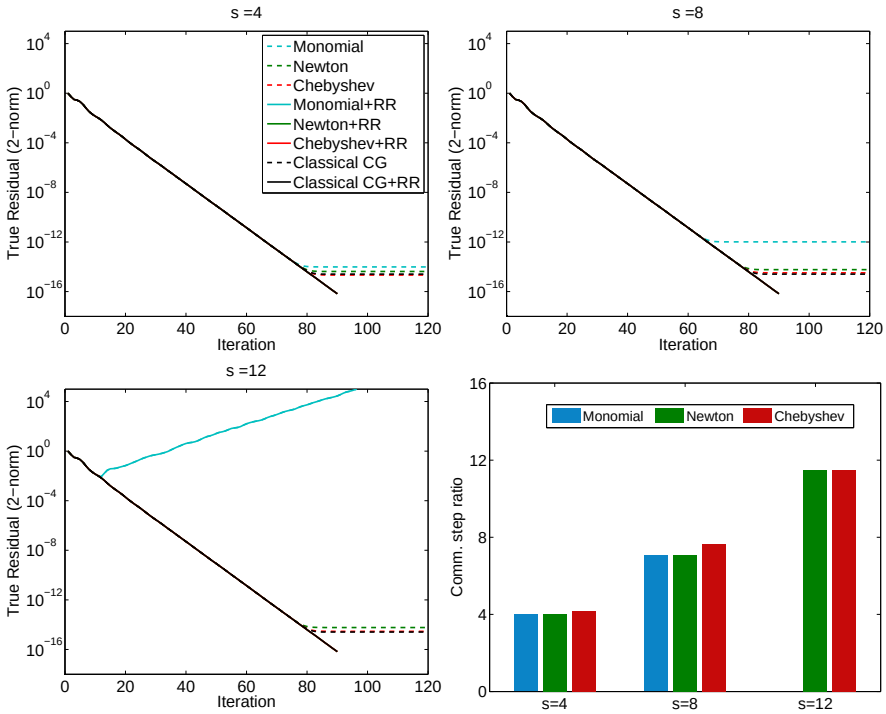

FIG. 5.3. *Convergence of true residual, FEM.*

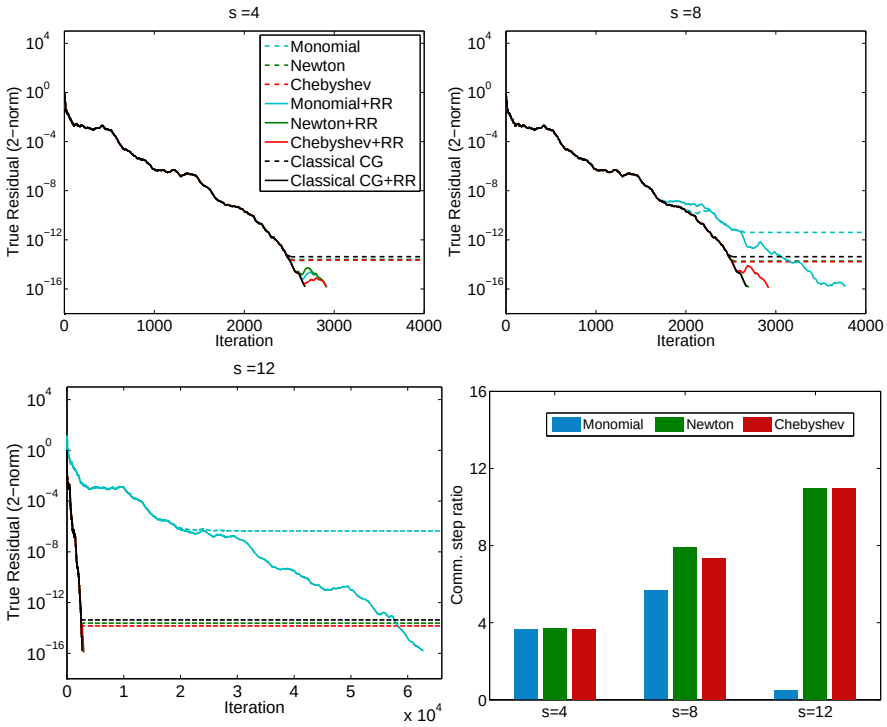

FIG. 5.4. *Convergence of true residual, circuit. Note the x-axis for* $s = 12$ *differs from* $s = [4, 8]$.

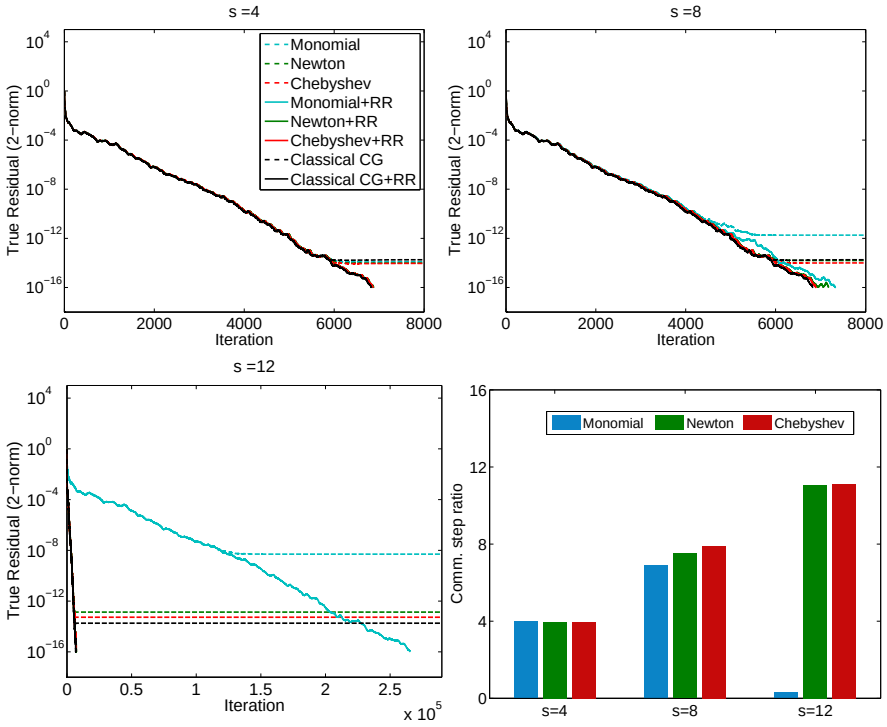

FIG. 5.5. *Convergence of true residual, ship. Note the x-axis for* $s = 12$ *differs from* $s = [4, 8]$.

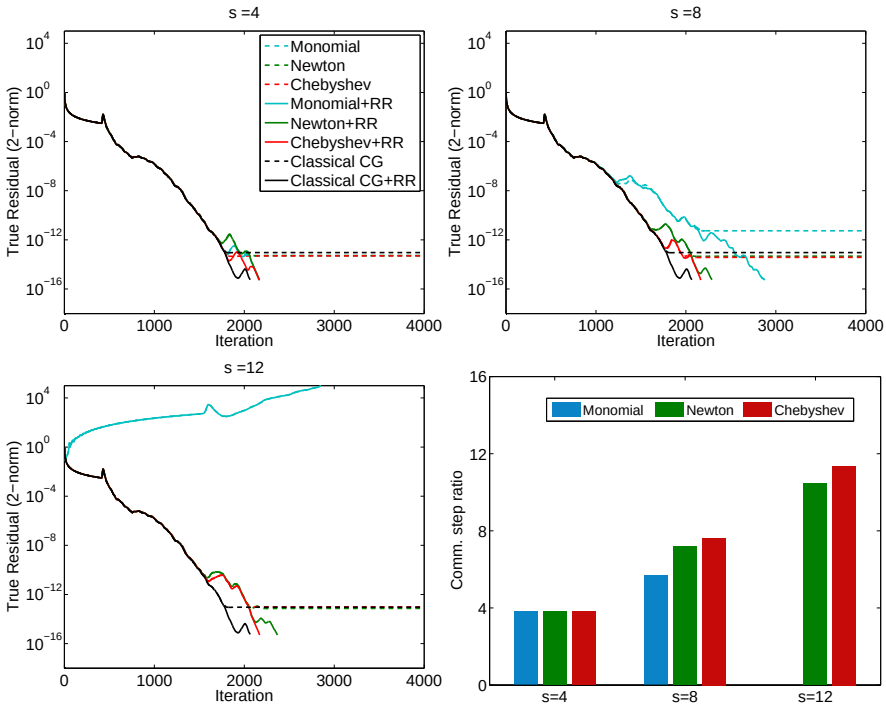

FIG. 5.6. *Convergence of true residual, thermal.*

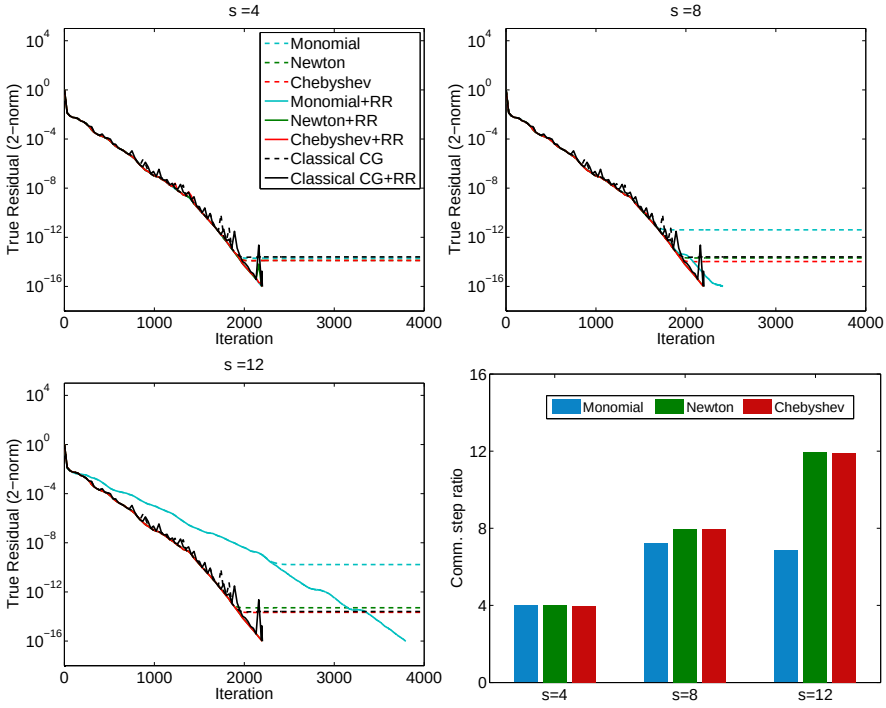

FIG. 5.7. *Convergence of true residual, xenon.*

5.1. Improvement in maximum attainable accuracy. Our results show that the residual replacement scheme for CA-KSMs succeeds in maintains an $O(\epsilon)$ agreement between the true and computed residuals for the monomial, Newton, and Chebyshev bases in both CA-CG and CA-BICG. For the matrices in Figs. 5.3 and 5.6, using the monomial basis with $s = 12$ causes divergence of the computed residual due to a numerically rank deficient Krylov basis. In these cases, because the computed residual does not converge, we do not expect that maintaining agreement between residuals will improve convergence of the true residual. Indeed, running these tests with the residual replacement strategy resulted in no residual replacements.

Furthermore, the convergence plots for all tests show that the residual replacement strategy also meets the second constraint of leaving the Lanczos process undisturbed. It is evident in all cases that the convergence rate of the CA-KSM is not decreased by the residual replacements steps. In Fig. 5.1, for the monomial basis with $s = 12$, we even see a slight increase in convergence rate using the residual replacement scheme.

In all cases, as shown in Table 5.2, the number of residual replacement steps is very small relative to the total number of iterations; we therefore claim that the additional communication cost of the residual replacement scheme (potentially an extra reduction for each replacement step) is negligible.

It is worth mentioning that if we have a very well-conditioned polynomial basis, the CA-KSM can achieve *higher accuracy* than the classical method (see, e.g.,the Newton and Chebyshev bases in Figs. 5.4 and 5.6). This is possible because our computations are now $O(s)$ rather than $O(n)$, which can result in slightly smaller local error.

5.2. Practical communication savings. The plots in the lower right of each figure show the communication step ratio, or the number of communication steps in the classical method divided by the CA variant, both with residual replacement (the same data is presented in Table 5.3). Ideally, for CA-CG and CA-BICG, this ratio should be around s; this would be the case if the CA method converges at around the same rate as the classical method with few required residual replacement steps. This is not performance data, but rather gives an upper bound on speedup we can expect from practical implementations – this upper bound would then be attained if we had a CA-KSM implementation that was $O(s)$ times faster than the classical method over s steps.

For the Newton and Chebyshev bases, we generally see the communication savings ratio increasing linearly with s, as we expect. This is because the Newton and Chebyshev bases remain well conditioned for higher s values (see, e.g., [2, 11, 19]), and thus the CA methods with these bases can maintain convergence similar to the classical method. Note that in Figs. 5.1, 5.2, 5.4, and 5.5, the Newton and Chebyshev data for $s = 12$ is less discernible, as the x-axis is compressed to show the slower but eventual convergence of the monomial basis; the behavior of the Newton and Chebyshev bases with $s = 12$ was similar to $s = 8$ in these cases, with only a modest increase in the number of iterations needed to converge (less than 9% in all cases, as can be deduced from Table 5.2).

For every test matrix, the CA-KSM with the monomial basis results shows an increase in communication savings ratio from $s = 4$ to $s = 8$, but the communication savings significantly decreases from $s = 8$ to $s = 12$ (for tests where the CA-KSM with $s = 12$ converges). For the test matrices shown in Figs. 5.1, 5.4 and 5.5, this decrease is so drastic that the resulting communication savings ratio is less than 1. Therefore, even if practical implementations of CA-KSMs achieve an $O(s)$ speedup

over s iterations versus the classical method, the run-time of the CA-KSM with the monomial basis for $s = 12$ will be *worse* than the classical method. In these cases, it is thus essential to use a well-conditioned basis such as Newton or Chebyshev, which still demonstrate $O(s)$ savings in communication for high s values.

This important observation indicates that, depending on the quality of the Krylov basis, a higher s value does not necessarily correspond to a faster algorithm or even less communication. For example, in our tests with the monomial basis, the convergence rate is slowed by finite precision error in computing the basis vectors for high values of s. Therefore, selecting a *lower* value of s which yields a *higher* convergence rate will result in fewer total communication steps, and thus faster run-time. It would therefore be useful to have an a priori estimate for the s value which maximizes the communication savings ratio. Developing heuristics to perform such an estimate based on properties of A and the computed Krylov basis is considered future work.

6. Conclusions and future work. In this work, we give the first analysis of maximum attainable accuracy in finite precision CA-KSMs. We bound the norm of the deviation of the true and computed residual and demonstrate that we can iteratively compute this bound within the CA-KSM for minimal additional cost. This computable bound enables an implicit residual replacement strategy which maintains the agreement of the true and computed residuals to $O(\epsilon)$. Numerical experiments demonstrate that if the finite precision computed residual converges, accuracy of order $O(\epsilon) \|A\| \|x\|$ can be achieved with a small number of residual replacement steps using the residual replacement scheme.

Much work remains to be done on the analysis of finite precision CA-KSMs. We plan to extend the analysis here to other CA-KSMs, such as CA-BICGSTAB, following the same general process. We note that we could also apply similar analyses to communication-avoiding variants of other recursively computed residual methods (see [9]).

We can also extend our error analysis to improve the convergence rate of CA-KSMs by taking into account error in the computation of the basis vectors and Lanczos coefficients. In other words, we could monitor when our computed residual deviates from what the true residual would be *in the classical algorithm*. For ill-conditioned bases, preliminary results indicate that this results in frequent restarts, often less than every s steps. This negates the benefits of using a higher s value.

We note that we could easily adapt our scheme to avoid breakdowns due to numerically rank-deficient bases by performing condition estimation on G^k in each outer loop. If a numerical rank deficiency is detected, we can schedule a residual replacement in the appropriate inner loop iteration. This will cause the method to begin a new outer loop and avoid computation with the degenerate basis vectors. We could also use this information to allow for the dynamic adjustment of s; if numerical rank deficiency is detected in a number of consecutive outer loop iterations, s could be decreased to reduce the basis condition number. Future work will investigate the effectiveness and practicality of this approach.

A primary direction for future work is heuristically determining the optimal s value based on properties of the matrix. As described above, the relationship between s and the total number of communication steps is no longer linear in finite precision arithmetic. The highest s value permitted by the structure of the matrix might require significantly more iterations to converge than a lower s value, resulting in more total communication steps and thus longer algorithm run-time.

REFERENCES

[1] Z. Bai, D. Hu, and L. Reichel, *A Newton basis GMRES implementation*, IMA J. Numer. Anal., 14 (1994), p. 563.

[2] E. Carson, N. Knight, and J. Demmel, *Avoiding communication in two-sided Krylov subspace methods*, Tech. Report UCB/EECS-2011-93, EECS Dept., U.C. Berkeley, Aug 2011. Submitted to SIAM J. Sci. Comp. special issue on the Twelfth Copper Mountain Conference on Iterative Methods.

[3] A. Chronopoulos and C. Gear, *On the efficient implementation of preconditioned s-step conjugate gradient methods on multiprocessors with memory hierarchy*, Parallel Comput., 11 (1989), pp. 37–53.

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

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

[6] T. Davis and Y. Hu, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), p. 1.

[7] J. Demmel, M. Hoemmen, M. Mohiyuddin, and K. Yelick, *Avoiding communication in computing Krylov subspaces*, Tech. Report UCB/EECS-2007-123, EECS Dept., U.C. Berkeley, Oct 2007.

[8] D. Gannon and J. Van Rosendale, *On the impact of communication complexity on the design of parallel numerical algorithms*, Trans. Comput., 100 (1984), pp. 1180–1194.

[9] A. Greenbaum, *Estimating the attainable accuracy of recursively computed residual methods*, SIAM J. Matrix Anal. Appl., 18 (1997), p. 535.

[10] A. Hindmarsh and H. Walker, *Note on a Householder implementation of the GMRES method*, tech. report, Lawrence Livermore National Lab.; Utah State Univ., Dept. of Mathematics, 1986.

[11] M. Hoemmen, *Communication-avoiding Krylov subspace methods*, PhD thesis, University of California, Berkeley, 2010.

[12] W. Joubert and G. Carey, *Parallelizable restarted iterative methods for nonsymmetric linear systems. Part I: Theory*, Int. J. Comput. Math., 44 (1992), pp. 243–267.

[13] C. Leiserson, S. Rao, and S. Toledo, *Efficient out-of-core algorithms for linear relaxation using blocking covers*, J. Comput. Syst. Sci. Int., 54 (1997), pp. 332–344.

[14] G. Meurant, *The Lanczos and conjugate gradient algorithms: from theory to finite precision computations*, vol. 19, Society for Industrial Mathematics, 2006.

[15] G. Meurant and Z. Strakos, *The Lanczos and conjugate gradient algorithms in finite precision arithmetic*, Acta Numer., 15 (2006), pp. 471–542.

[16] M. Mohiyuddin, M. Hoemmen, J. Demmel, and K. Yelick, *Minimizing communication in sparse matrix solvers*, in Proc. Conf. on High Perf. Comp. Net., Storage and Anal., ACM, 2009, p. 36.

[17] A. Neumaier. Oral presentation. Numerical Linear Algebra Meeting, Oberwolfach, 1994.

[18] C. Paige, M. Rozioznik, and Z. Strakos, *Modified Gram-Schmidt (MGS), least squares, and backward stability of MGS-GMRES*, SIAM J. Matrix Anal. Appl., 28 (2006), p. 264.

[19] B. Philippe and L. Reichel, *On the generation of Krylov subspace bases*, Appl. Numer. Math, 62 (2012), pp. 1171–1186.

[20] G. Sleijpen and H. van der Vorst, *Reliable updated residuals in hybrid Bi-CG methods*, Computing, 56 (1996), pp. 141–163.

[21] E. Sturler, *A performance model for Krylov subspace methods on mesh-based parallel computers*, Parallel Comput., 22 (1996), pp. 57–74.

[22] S. Toledo, *Quantitative performance modeling of scientific computations and creating locality in numerical algorithms*, PhD thesis, MIT, 1995.

[23] C. Tong and Q. Ye, *Analysis of the finite precision bi-conjugate gradient algorithm for nonsymmetric linear systems*, Math. Comp., 69 (2000), pp. 1559–1576.

[24] 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 (1999), pp. 835–852.

[25] J. Van Rosendale, *Minimizing inner product data dependencies in conjugate gradient iteration*, Tech. Report 172178, ICASE-NASA, 1983.

[26] H. Walker, *Implementation of the GMRES method using Householder transformations*, SIAM J. Sci. Stat. Comput., 9 (1988), p. 152.