B639388 Subcontract Award LLNL-Charles University: Mixed Precision Numerical Linear Algebra

Quarter 2 Report: Numerical Experiments and Identification of Areas for Further Consideration

Erin Carson and Tomáš Gergelits

1 Summary of activities

The second quarter of the project was spent carrying out numerical experiments on a variety of test matrices, using a combination of precisions, with the intent to study the numerical properties (namely, accuracy and convergence behavior) of the Conjugate Gradient (CG) method. We summarize our findings in the remainder of the document. Other activities include attending biweekly xSDK meetings.

We are also currently collaborating with Steven Thomas (NREL) on the numerical stability analysis of "low-synch" Gram-Schmidt routines, for potential application within high-performance GMRES variants. This work is in progress.

The subsequent quarter will be spent delving into the numerical analysis in order to provide theoretical explanation for the behavior observed in our experiments.

2 Numerical Experiments

We begin our investigation with a study of the behavior of the selective use of low precision in the basic CG method. We will use the Hestenes and Stiefel variant of CG [4], which we call HSCG. A mixed precision variant of this algorithm is shown in Algorithm 1. Within this algorithm, ε represents the working precision (the default precision as well as the precision in which quantities are stored), ε_{IP} represents the precision used to compute the inner products, and ε_{MV} represents the precision used to compute the matrix-vector products. For simplicity, we assume here an initial value of 0 for the approximate solution vector.

Algorithm 1 Hestenes and Stiefel CG (HSCG)

Input: $N \times N$ symmetric matrix A, length-N vector b, number of steps k
Output: Approximate solution x_{k+1}

1: $r_0 = b$, $p_0 = b$, $x_0 = 0$
2: **for** $i = 0, 1, \ldots, k$ **do**
3: $\quad \alpha_i = r_i^T r_i / p_i^T A p_i$
4: $\quad x_{i+1} = x_i + \alpha_i p_i$
5: $\quad r_{i+1} = r_i - \alpha_i A p_i$
6: $\quad \beta_{i+1} = r_{i+1}^T r_{i+1} / r_i^T r_i$
7: $\quad p_{i+1} = r_{i+1} + \beta_{i+1} p_i$
8: **end for**

We consider three different mixed precision approaches. In all approaches we assume that the data is stored and vector computations (axpys) are performed in the given working precision. In one approach, the SpMV computation is performed in a lower precision ε_{MV}. In the next approach, the inner products are all performed in a lower precision ε_{IP}. In another approach, both the SpMV and the inner products are done in the same lower precision, $\varepsilon_{MV} = \varepsilon_{IP}$.

We first study a class of matrices which is commonly used in studies of the behavior of CG (see, e.g., the book [5]. This is a class of diagonal matrices of dimension N with eigenvalues

$$\lambda_i = \lambda_1 + \left(\frac{i-1}{N-1}\right)(\lambda_N - \lambda_1)\rho^{N-i}, \quad i = 1, \ldots, N, \tag{2.1}$$

where the parameters λ_1 and λ_N are the smallest and largest eigenvalues, respectively, and ρ controls the distribution of the eigenvalues. For small ρ, this gives a cluster at the lower end of the spectrum. As ρ approaches 1, the eigenvalues become more equally spaced. To demonstrate this, in Figure 1 we plot the spectra for $\rho = \{.4, .65, .9\}$ (the values used in the experiments presented here) for a matrix with $N = 40$, $\lambda_1 = 0.1$, and $\lambda_N = 100$.

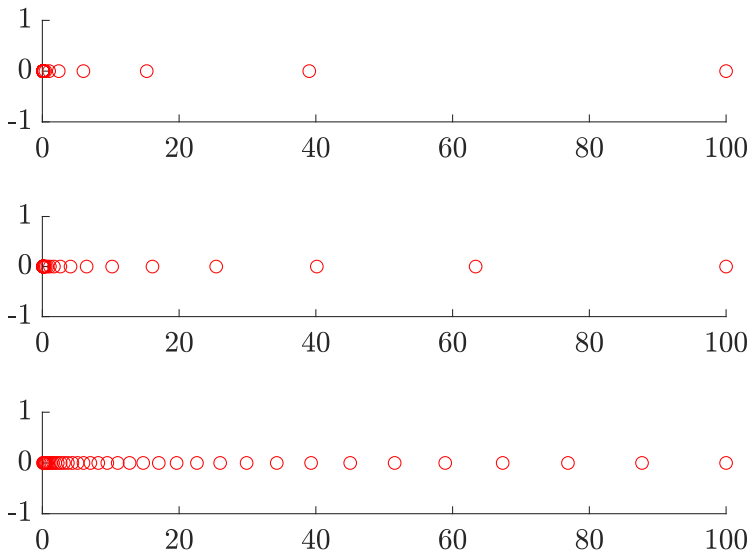

Figure 1: Eigenvalues of matrices generated by (2.1) with $N = 40$, $\lambda_1 = 0.1$, $\lambda_N = 100$, and $\rho = \{.4, .65, .9\}$ (top, middle, and bottom plots, respectively).

We have thoroughly tested a number of different parameter combinations; we report the most interesting and relevant results here. Our goal is to determine in what way the numerical behavior using various precisions depends on the numerical properties of the coefficient matrix and right-hand side. In the following three subsections, we test right-hand sides constructed in three different ways. For each of these constructions, we use the matrix described above with the parameters $N = 40$, $\rho \in \{.4, .65, .9\}$ and $\kappa(A) \in \{10^1, 10^3, 10^6, 10^9\}$, where $\kappa(A) = \lambda_N/\lambda_1$ is the 2-norm condition number of A and we used $\lambda_1 = 10^{-1}$. Note that we tested other values of λ_1, but this did not have any significant impact on numerical behavior we observed. All tests are performed in MATLAB version 9.8.0.1451342 (R2020a) Update 5. We use the built-in MATLAB single and double datatypes, as well as the FP16 functionality through the Cleve Lab MATLAB App function 'vfp16'. We measure the numerical behavior by plotting the relative A-norm of the error, defined as $\|x_i - x\|_A/\|x\|_A = \sqrt{(x_i - x)^T A(x_i - x)}/\sqrt{x^T Ax}$, where x is the true solution obtained via higher precision LU factorization (via the Advanpix Toolbox).

2.1 Right-hand sides with equal components in the eigenspace of A

In this subsection, we use right-hand sides constructed such that they contain equal components in the eigenspace of A. This should represent a case that is somewhat of a worst-case scenario for convergence, as the entire eigenspace must be explored. Results for $\rho = .4, .65, .9$ are shown in Figures 2, 3, and 4, respectively.

We can notice a few things from these experiments. First, if we perform all inner products in single precision and everything else in double precision, we can still obtain accuracy to the working precision (double). This is not true for the other mixed precision variants, e.g., the algorithms where the SpMVs are done in lower precision; in these cases, single precision MVs result in accuracy on the order of single precision. The second thing to notice is that the convergence delay increases as the conditioning of the problem increases. This is true for all variants in which some part (inner products or matrix-vector products or both) are performed in the lower precision. We expect that this is because perturbations on the order of single precision are introduced into the Lanczos recurrence, whether they come from MVs or inner products. This effect is yet to be theoretically studied but we investigate further in Section 3.

Another thing to point out is the different behavior for different eigenvalue distributions. When eigenvalues are clustered in the lower end of the spectrum, convergence of exact CG is relatively fast, and the same can be said for all finite precision variants when the system is well conditioned. For exact CG for a fixed condition number, the number of iterations required to converge generally increases as ρ increases. We might expect the number of iterations required for the finite precision versions to also get worse as ρ increases, but this is not necessarily the case. For example, compare the plots with condition numbers $\kappa(A) = 10^6$ and $\kappa(A) = 10^9$ (the lower left and right plots) in Figures 3 (with $\rho = .65$) and 4 (with $\rho = .9$). In both cases, the convergence delay of all finite precision variants is worse for $\rho = .65$ than it is for $\rho = .9$. This effect must be investigated. It is clear that something goes drastically wrong when $\rho = .9$ and $\kappa(A) = 10^9$ and everything is computed in single precision, since the norm of the error is nonmonotonically decreasing.

We wonder whether or not this behavior, namely, that double precision accuracy is attainable even when

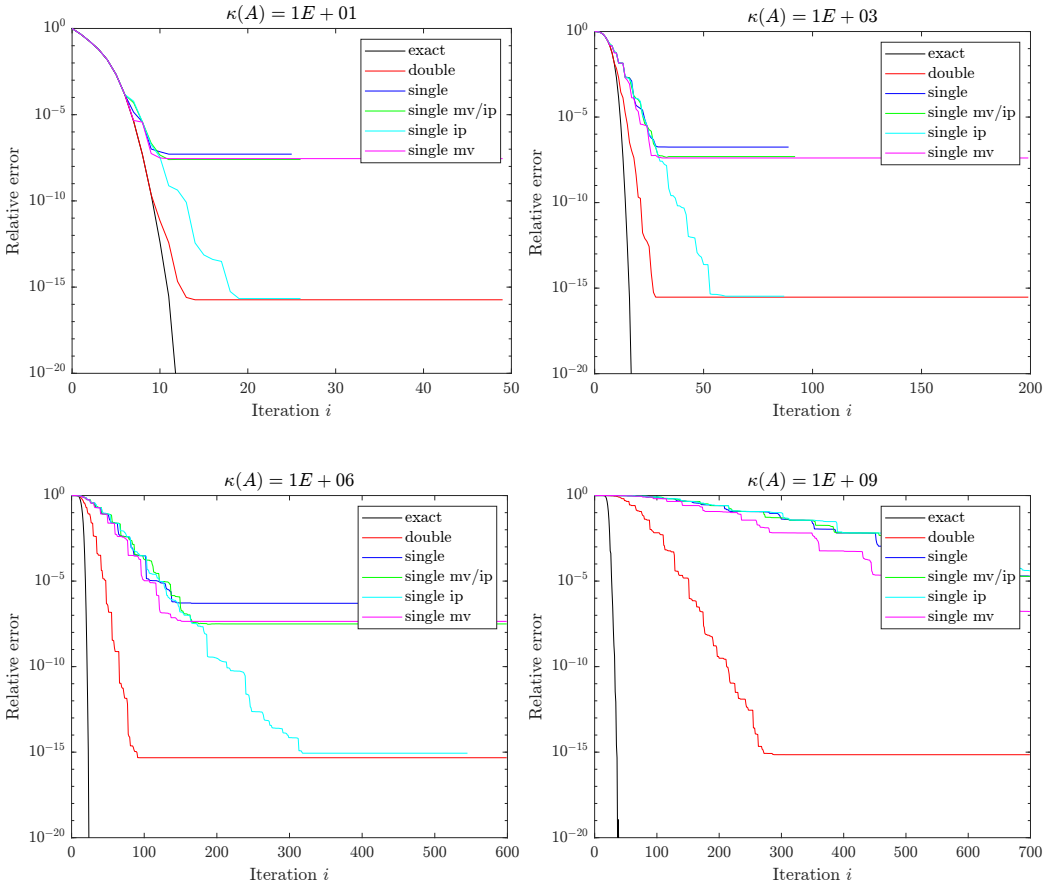

Figure 2: Right-hand sides with equal components in the eigenspace of A, $\rho = .4$

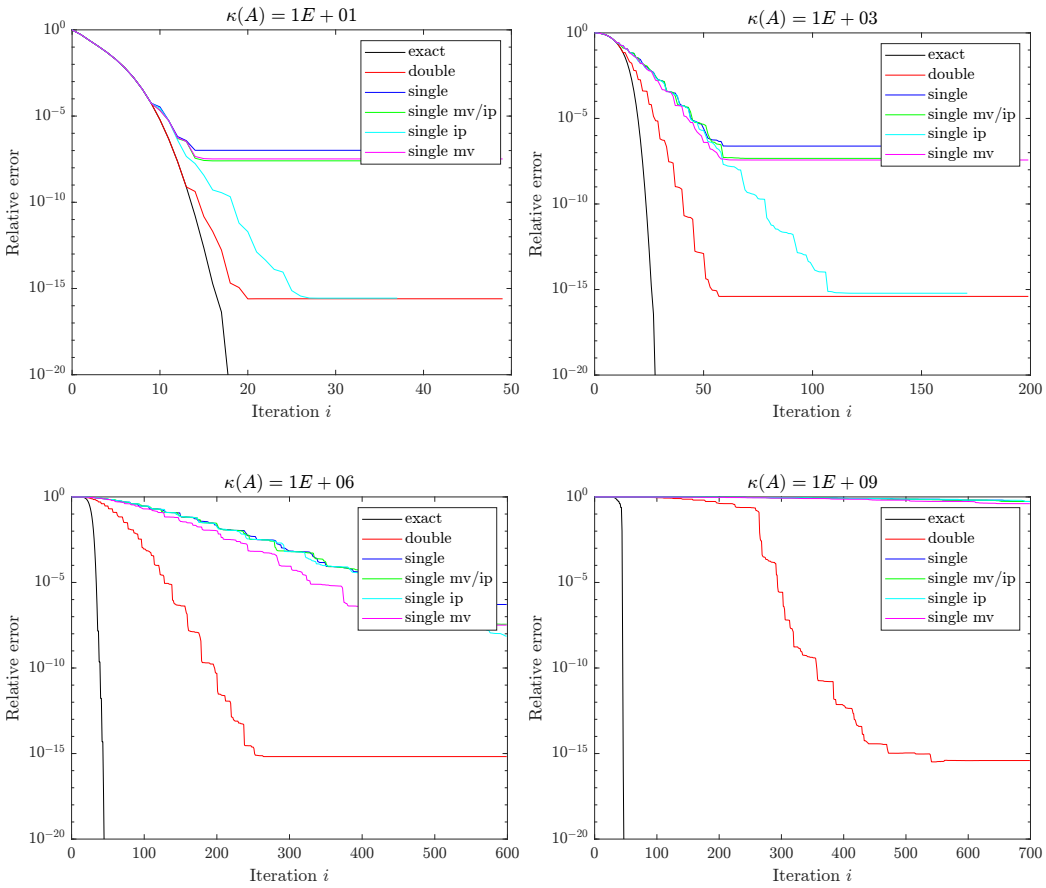

Figure 3: Right-hand sides with equal components in the eigenspace of A, $\rho = .65$

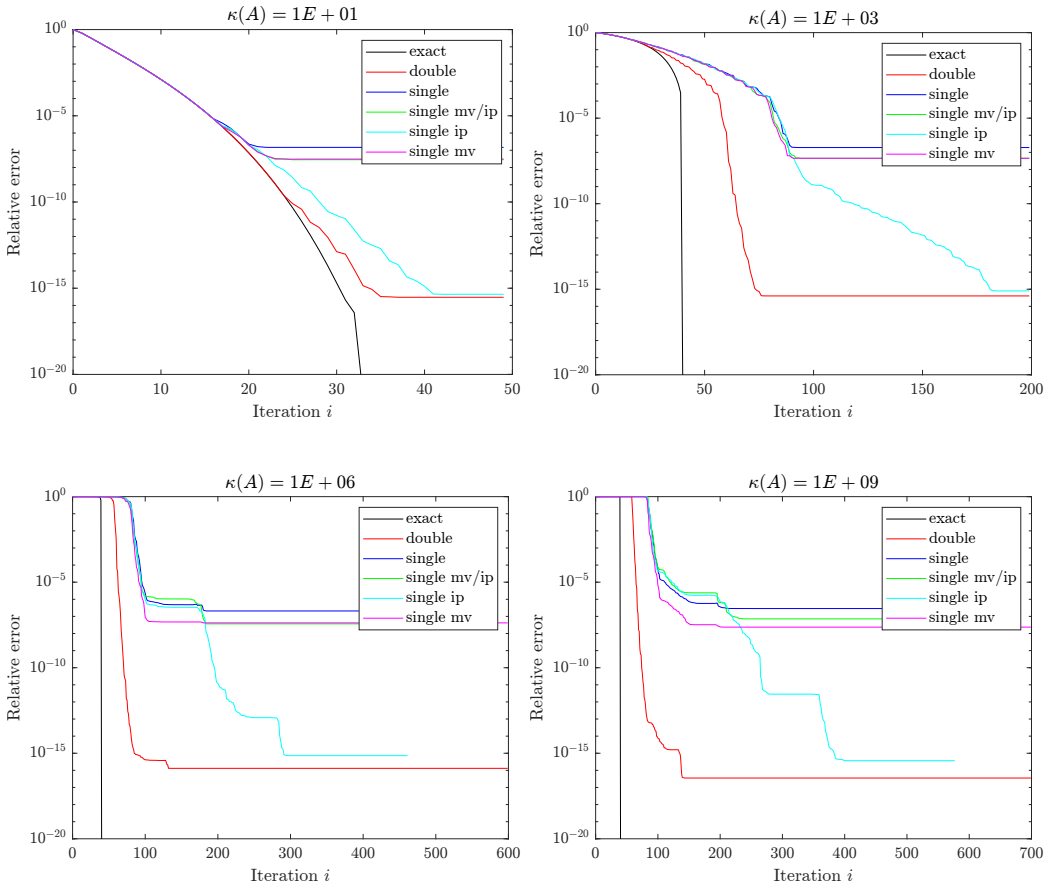

Figure 4: Right-hand sides with equal components in the eigenspace of A, $\rho = .9$

inner products are computed in lower precision, persists when 16-bit precision is used in the inner products. In Figures 5, 6, and 7, we repeat the experiments in Figures 2, 3, and 4, respectively, testing both inner products in FP16 and bfloat16 formats as well. Both FP16 and bfloat16 use 16 bits total, but the distribution of bits in the representation of numbers differs. In IEEE FP16, there is 1 sign bit, 5 exponent bits, and 10 explicitly-stored significand bits. The bfloat16 format in contrast uses 1 sign bit, 8 exponent bits, and 7 explicitly-stored significand bits. In other words, bfloat16 uses the same number of total bits as half precision, but the same number of exponent bits as single precision, with the bits being borrowed from the significand. The result is that the range of representable numbers is greater than that of half precision, but the number of significant digits representable is less than that of half precision, with a unit roundoff of $\varepsilon = 2^{-8} \approx 3.9 \times 10^{-3}$.

We notice that when bfloat16 is used for the inner products, we can, as with single precision inner products, obtain double precision accuracy when the method is convergent. Of course, since the unit roundoff is larger for bfloat16, the convergence is even further delayed compared with inner products in single precision; in some cases this is so extreme as to make such an approach impractical. For example, in Figure 5 for $\kappa(A) = 10^6$, the variant 'bfloat16 ip' *is* converging, but so slowly that it has only reached a relative error of around 10^{-2} after 600 iterations; in contrast, 'single ip' converges to double precision accuracy after about 300 iterations (which is about 3× more iterations than CG in double precision).

We also notice that for these problems, when inner products are computed in FP16, the method often breaks down once the relative accuracy reaches around 10^{-4}. We checked the approximate solution vectors at this point, and around this point they contain NaN quantities; this is due to the limited range representable by FP16. However, up until this point, 'fp16 ip' generally converges faster than 'bfloat16 ip', which is due to the lower unit roundoff in this case.

We can explain theoretically why high accuracy is still attainable despite lower precision inner products by extending the work of Greenbaum on bounding the maximum attainable accuracy to the mixed precision case [3]. We outline this approach in the subsection below; a full analysis remains future work.

2.1.1 Theoretical analysis of attainable accuracy in mixed precision CG

We assume all computations in CG are done and data stored in working precision ϵ, except for inner products which are performed in precision ϵ_{IP} and SpMVs which are performed in precision ϵ_{MV}. We sketch a proof of the maximum attainable accuracy for this mixed precision case.

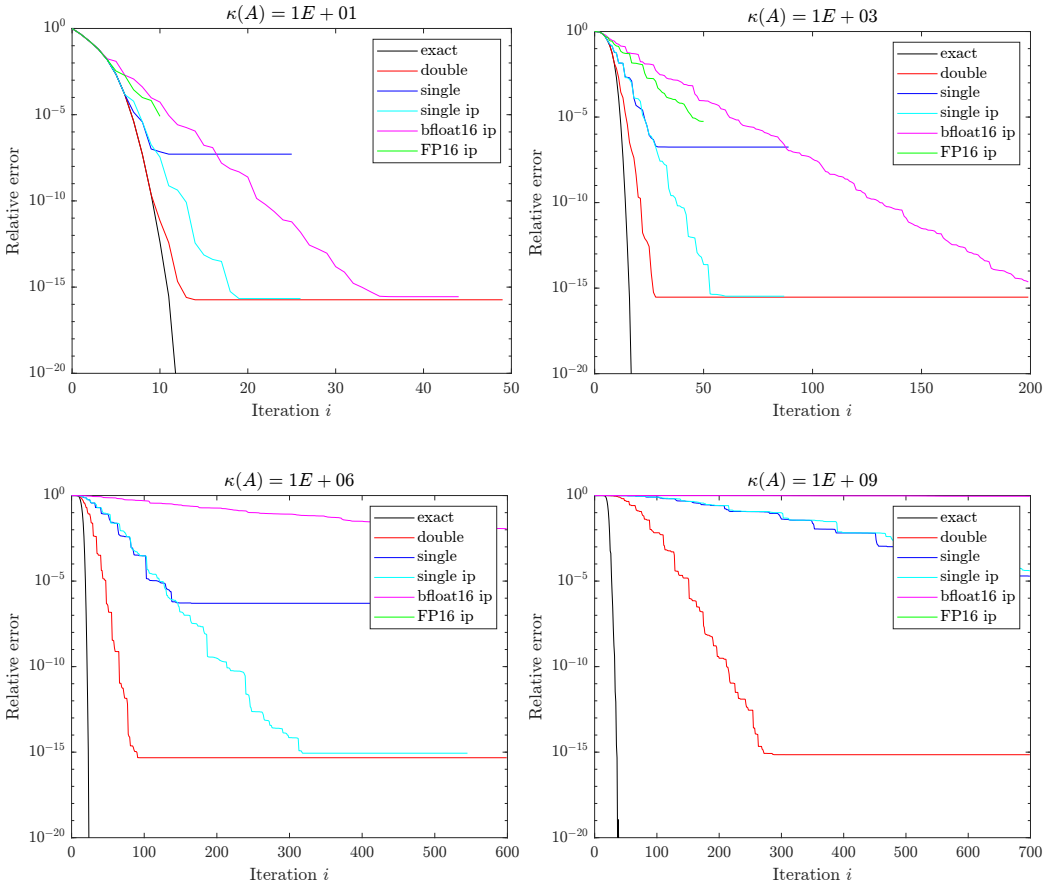

Figure 5: Half precision, $\rho = .4$

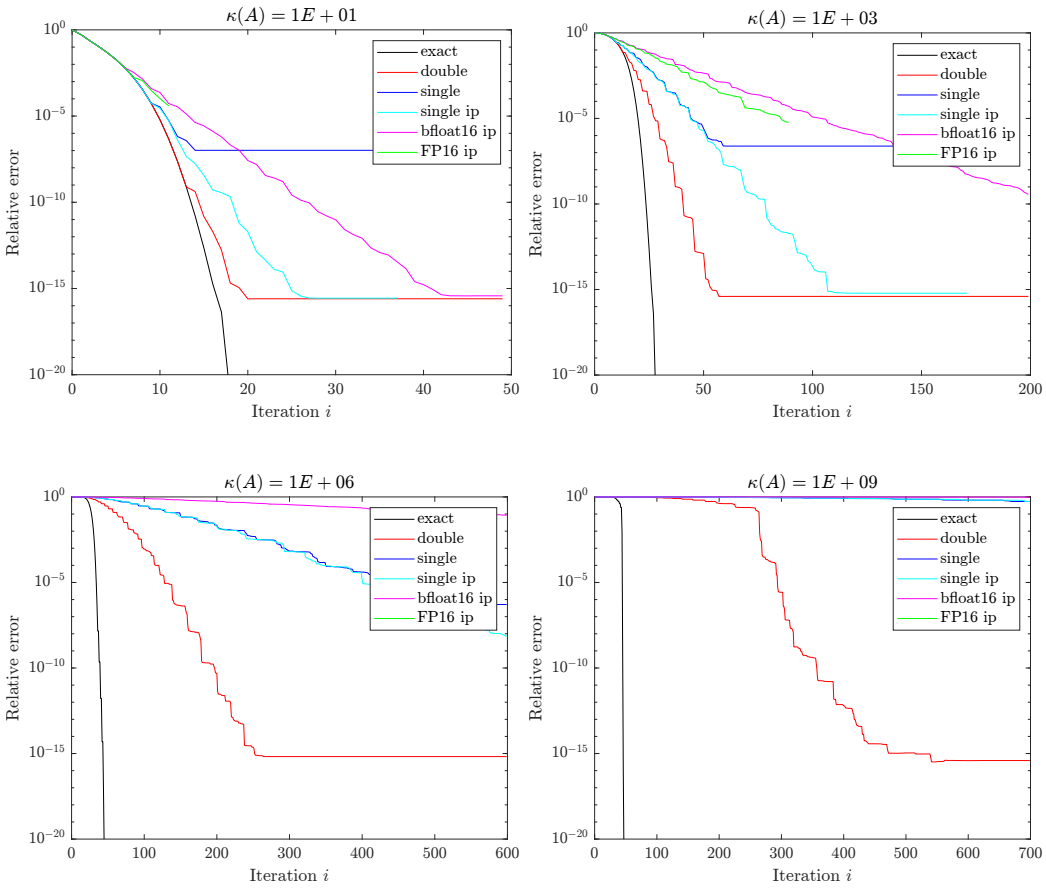

Figure 6: Half precision, $\rho = .65$

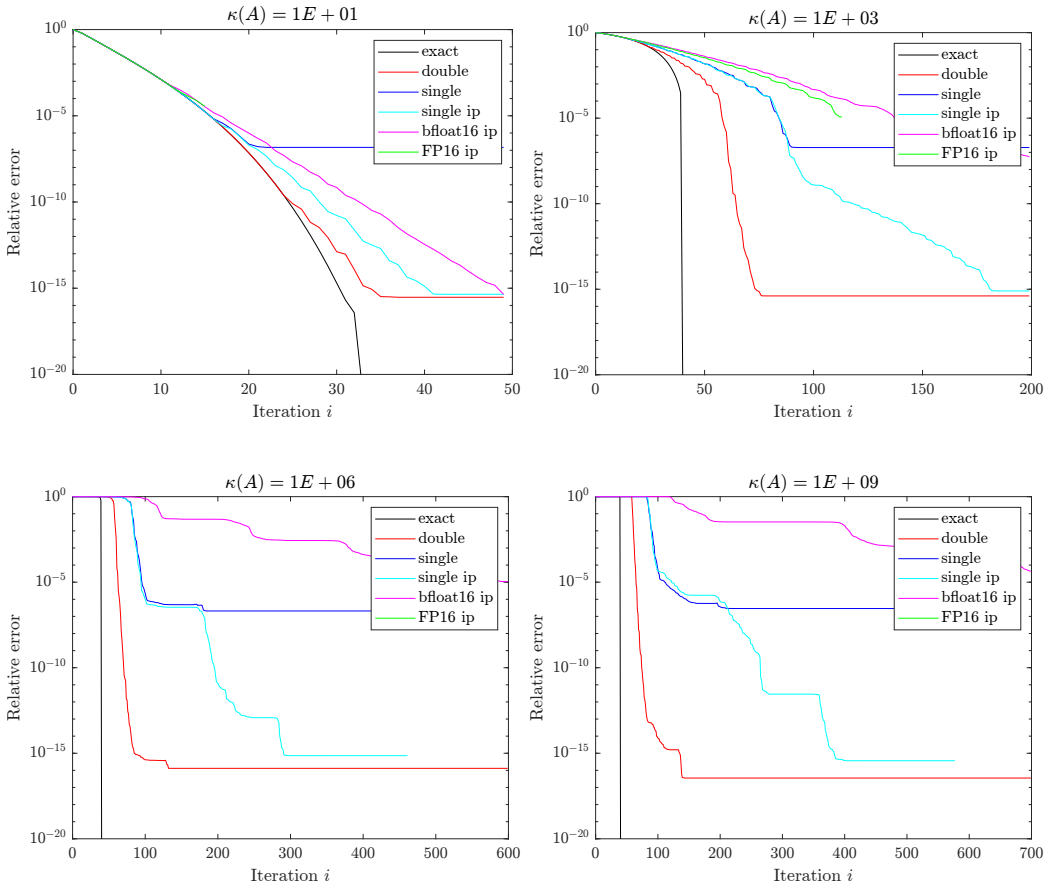

Figure 7: Half precision, $\rho = .9$

We use standard rounding error analysis to bound the following errors, where A is an $N \times N$ matrix with at most m nonzeros per row, α is a scalar, and v and w are length-N vectors.

$$\begin{aligned} fl(\alpha v) &= \alpha v + \delta_1, & \|\delta_1\| &\leq \varepsilon \|\alpha v\| \\ fl(v + w) &= v + w + \delta_2, & \|\delta_2\| &\leq \varepsilon(\|v\| + \|w\|) \\ fl(v^T w) &= v^T w + \delta_3, & \|\delta_3\| &\leq N(\varepsilon_{IP} + O(\varepsilon_{IP}^2))\|v\|\|w\| \\ fl(Av) &= Av + \delta_4, & \|\delta_4\| &\leq mN^{1/2}\varepsilon_{MV}\|A\|\|v\| \end{aligned}$$

We can write the true residual $b - Ax_k = b - Ax_k - r_k + r_k$, and thus

$$\|b - Ax_k\| \leq \|b - Ax_k - r_k\| + \|r_k\|.$$

Assuming that the method converges (which is another issue to be investigated), we will eventually have $\|r_k\| \to 0$, and thus the true residual (related to the accuracy) will be limited by the size of the residual gap $\|b - Ax_k - r_k\|$.

Following basic arithmetic, it can be shown that this gap can be bounded by

$$\frac{\|b - Ax_k - r_k\|}{\|A\|\|x\|} \leq \frac{\|b - Ax_0 - r_0\|}{\|A\|\|x\|} + k\left(\varepsilon + (6\varepsilon + mN^{1/2}\varepsilon_{MV})\Theta_k\right),$$

where $\Theta_k = \max_{j \leq k} \|x_j\|/\|x\|$. There are two important things to notice. First, the quantity ε_{IP} appears nowhere in this bound. Thus, assuming that the precisions used are such that the method converges, the precision used in the inner products does not contribute to the residual gap and thus does not harm the accuracy. Second, assuming that $\varepsilon_{MV} \geq \varepsilon$, the error from the SpMV will dominate this bound. Thus doing matrix-vector products in lower than the working precision can potentially have an affect on the accuracy. We expect that similar results can be obtained for mixed precision variants of other recursively-computed residual methods such as BiCGSTAB [7].

We note that whether or not the method still converges using the specified precisions is an important point and is non-obvious; further investigation in this area is needed.

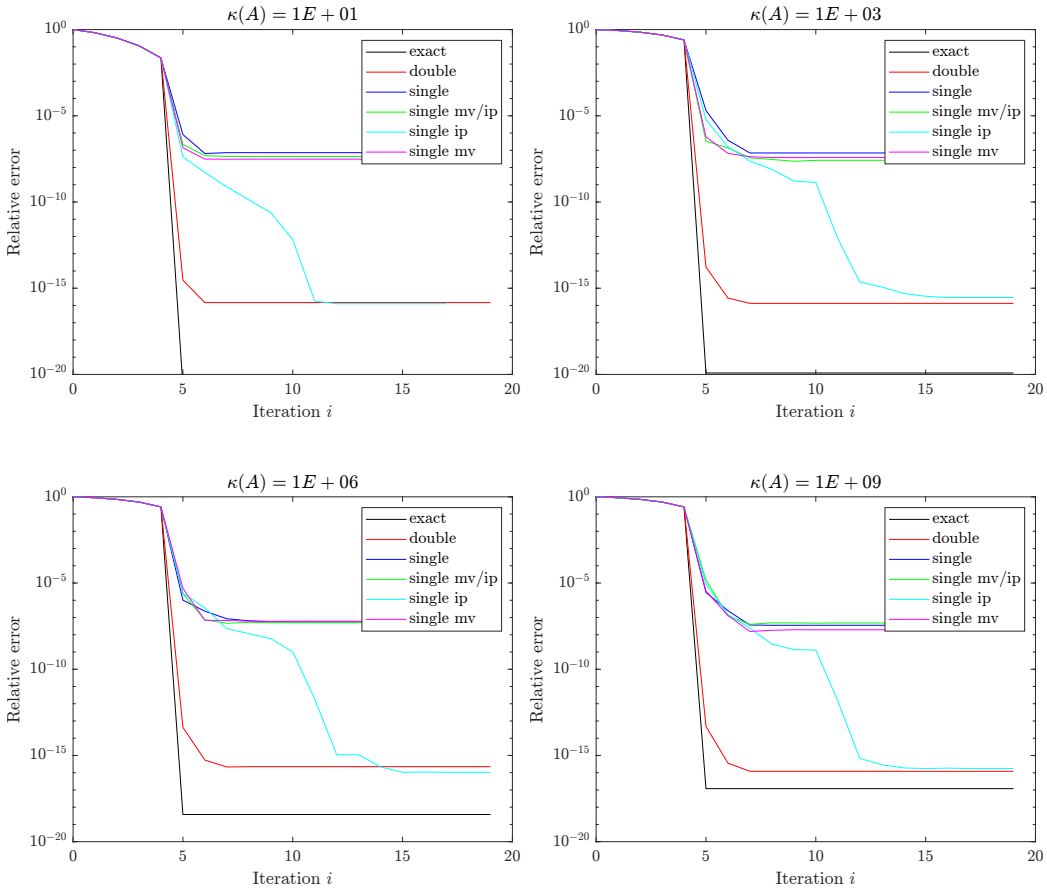

Figure 8: Right-hand sides with components in the eigenspace of A corresponding to small eigenvalues, $\rho = .4$

2.2 Right-hand sides with components in the eigenspace of A corresponding to small eigenvalues

In this subsection, we test the same matrices, but with right-hand sides constructed such that they contain components only in the eigenspace of A corresponding to the 5 smallest eigenvalues. Results for $\rho = .4, .65, .9$ are shown in Figures 8, 9, and 10, respectively.

This is a much easier problem for the CG method. The upper part of the spectrum has no influence on the behavior of the method, as if all but the 5 smallest eigenvalues had been deflated. Thus exact CG converges in 5 iterations. The finite precision behavior also reflects this. The convergence behavior of all finite precision variants is quite close to that of exact CG, even for the very ill-conditioned systems. Even the 'single ip' CG variant converges in 12 iterations in the worst case, even when $\kappa(A) = 10^9$. This example emphasizes the importance of the right-hand side in determining convergence behavior, and serves as a caution against making any predictions about the numerical behavior of CG based on $\kappa(A)$ alone. Thus any method for determining which combination of precisions should be used will need to take into account both A and b.

These general results also hold for even lower precision computations. In Figure 11, we run the same problems as in Figure 10 (with $\rho = .9$) but test the 16-bit inner product variants. In particular, 'bfloat16 ip' seems promising; double precision accuracy is attainable and the number of iterations required versus double precision is about a factor of 3 even for very ill-conditioned problems.

2.3 Problems with set solution

In this subsection, we test the same matrices, but we set the elements of the right-hand side b to be $b_i = (-1)^{i+1} A_{i,i}$. This gives a true solution of $x = [-1, 1, -1, 1, \ldots]^T$. Results for $\rho = .4, .65, .9$ are shown in Figures 12, 13, and 14, respectively. Our observations here are generally the same as in Section 2.1. Computing inner products in lower precision does not significantly affect the accuracy, but can significantly increase the number of iterations until convergence, especially for more ill-conditioned problems. Again, the matrices with $\rho = .65$ seem to have the worst convergence behavior for finite precision variants.

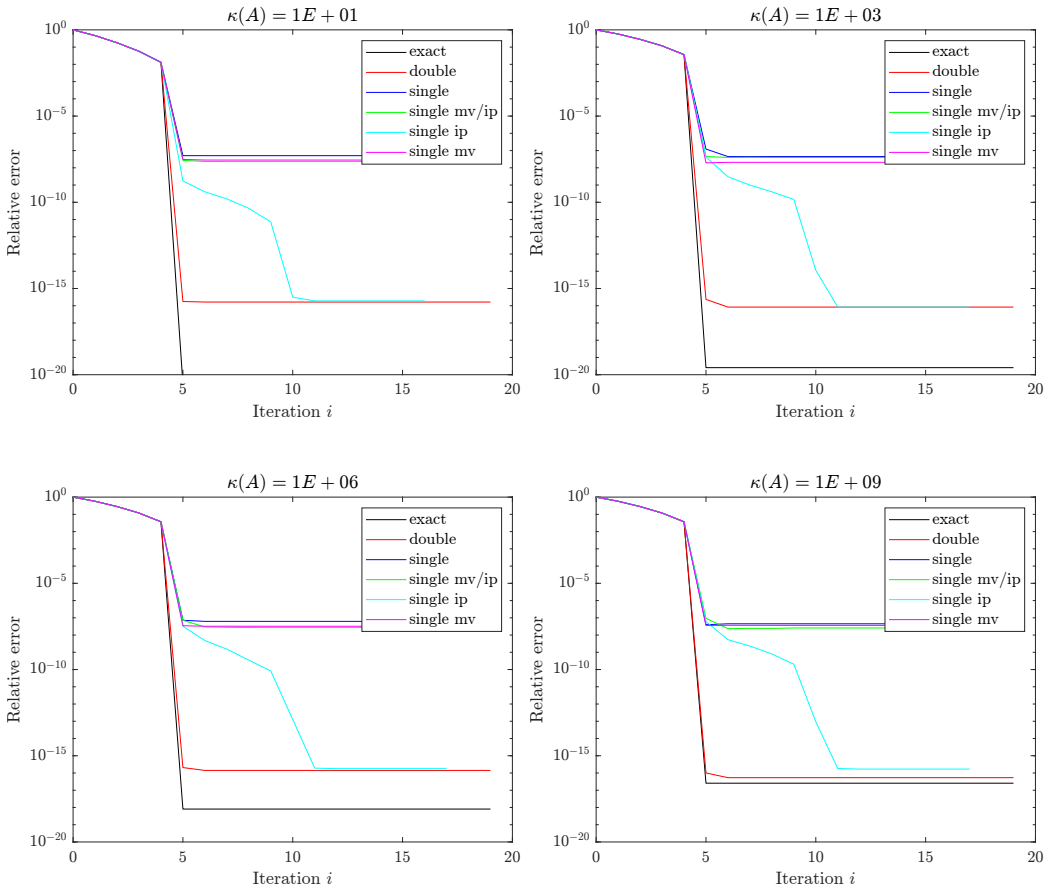

Figure 9: Right-hand sides with components in the eigenspace of A corresponding to small eigenvalues, $\rho = .65$

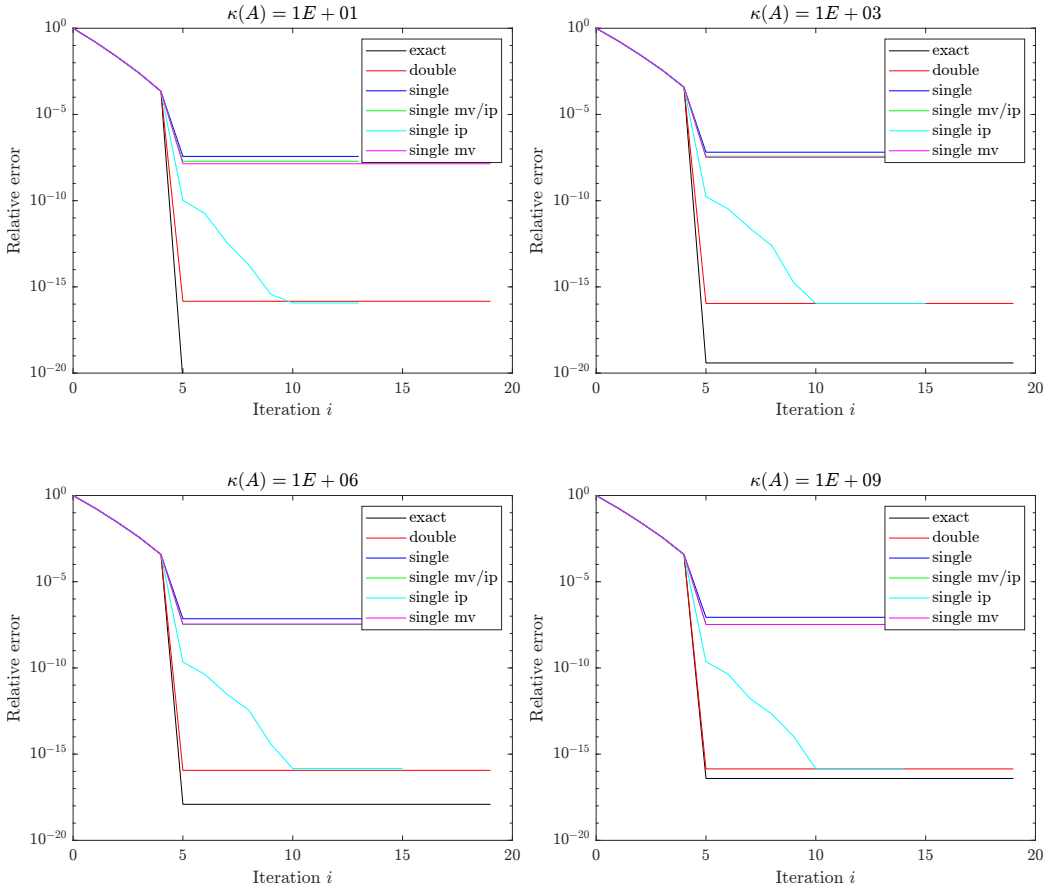

Figure 10: Right-hand sides with components in the eigenspace of A corresponding to small eigenvalues, $\rho = .9$

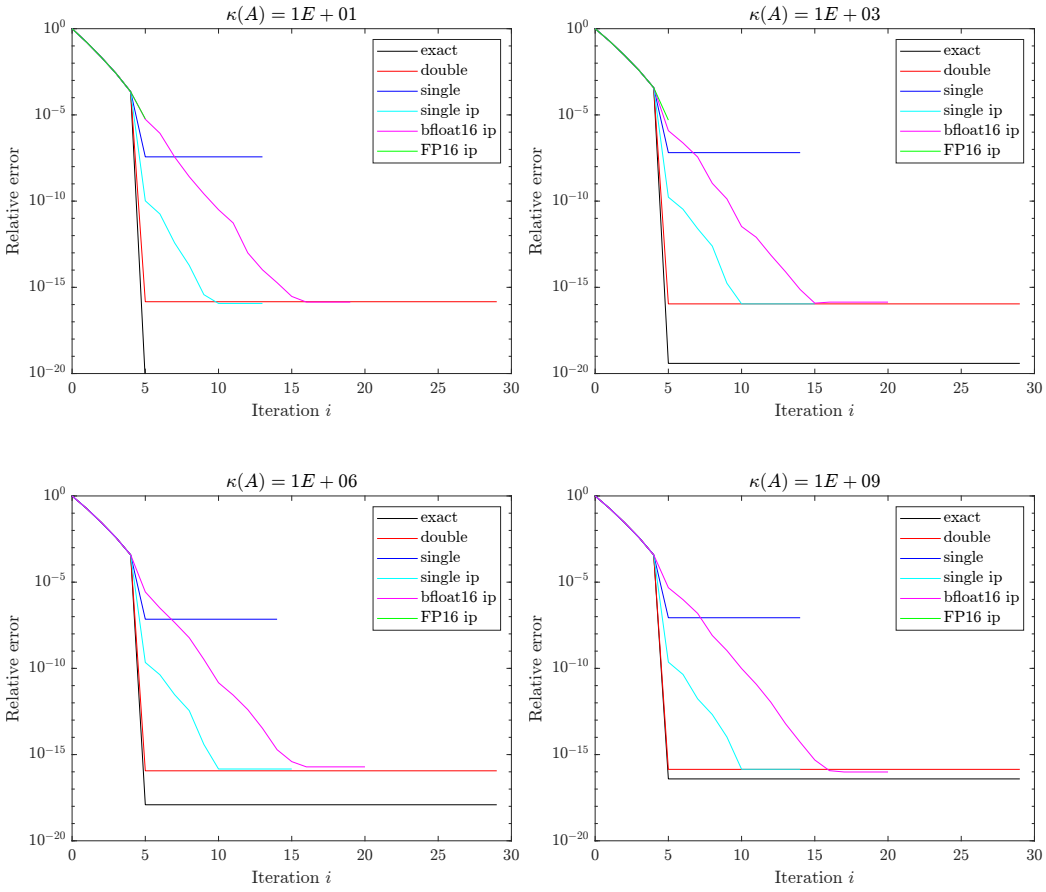

Figure 11: Right-hand sides with components in the eigenspace of A corresponding to small eigenvalues with half precision variants, $\rho = .9$

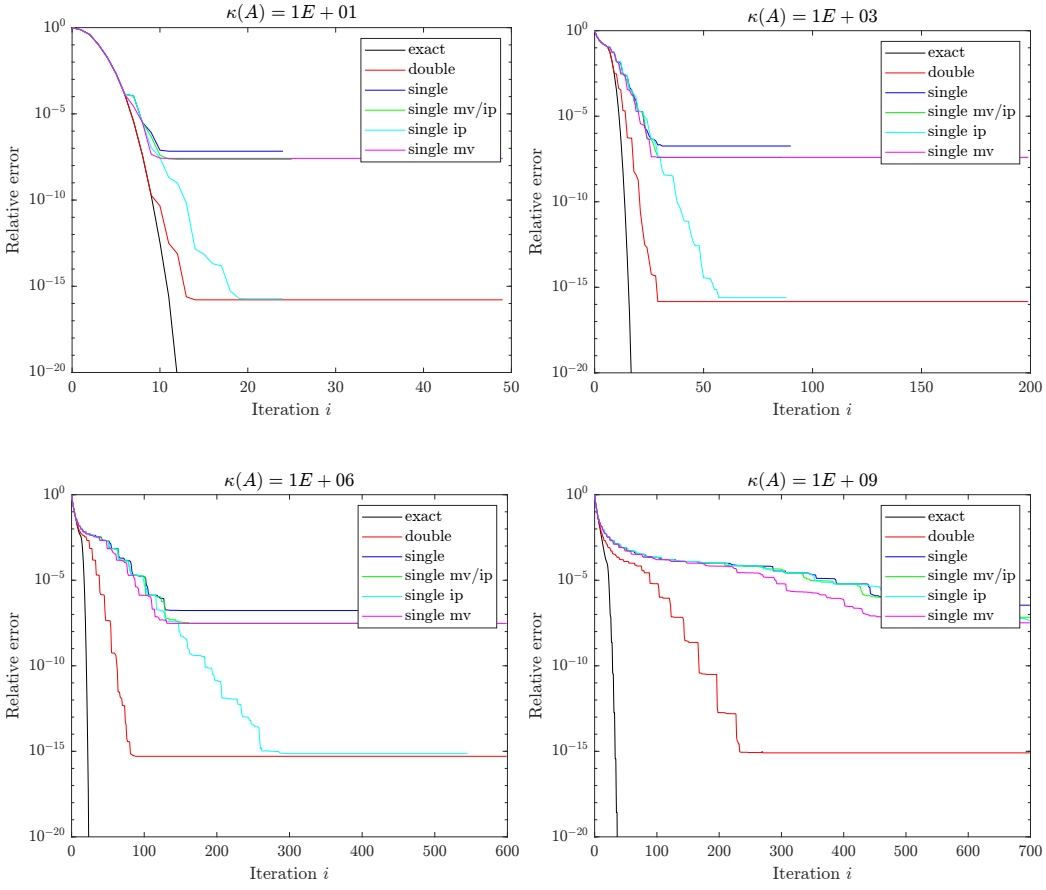

Figure 12: Problems with set solution, $\rho = .4$

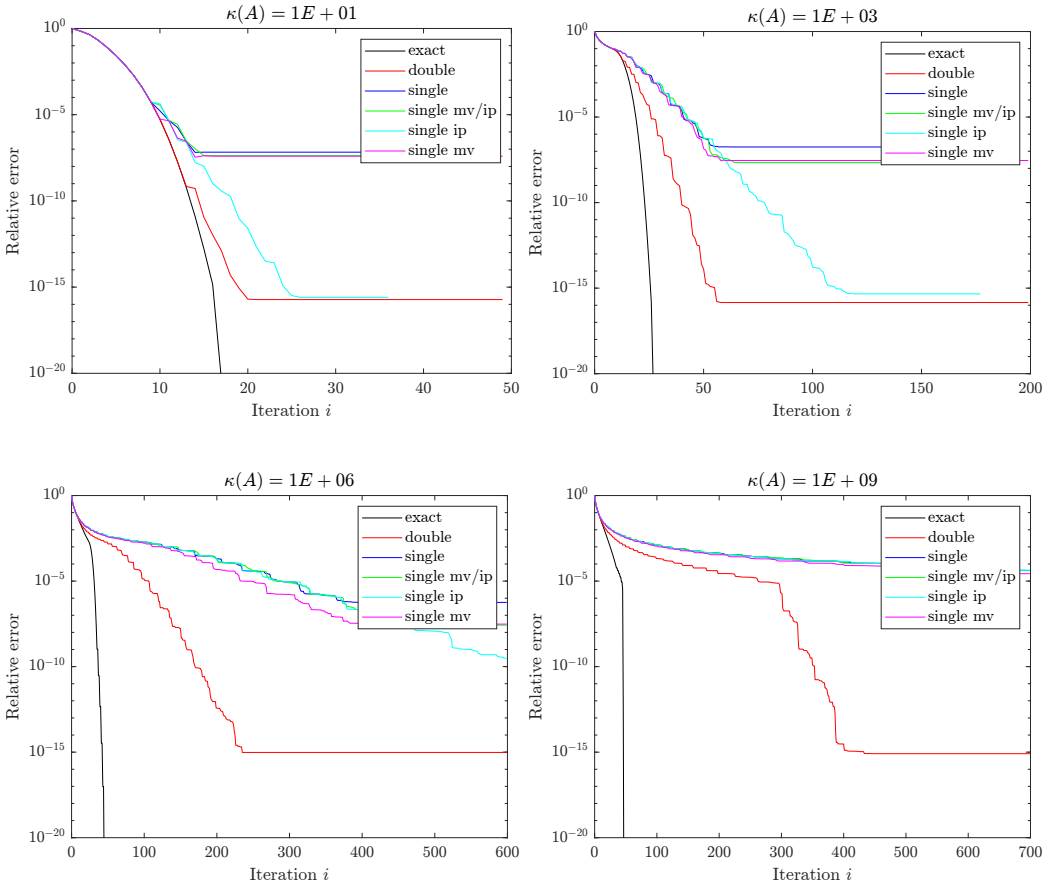

Figure 13: Problems with set solution, $\rho = .65$

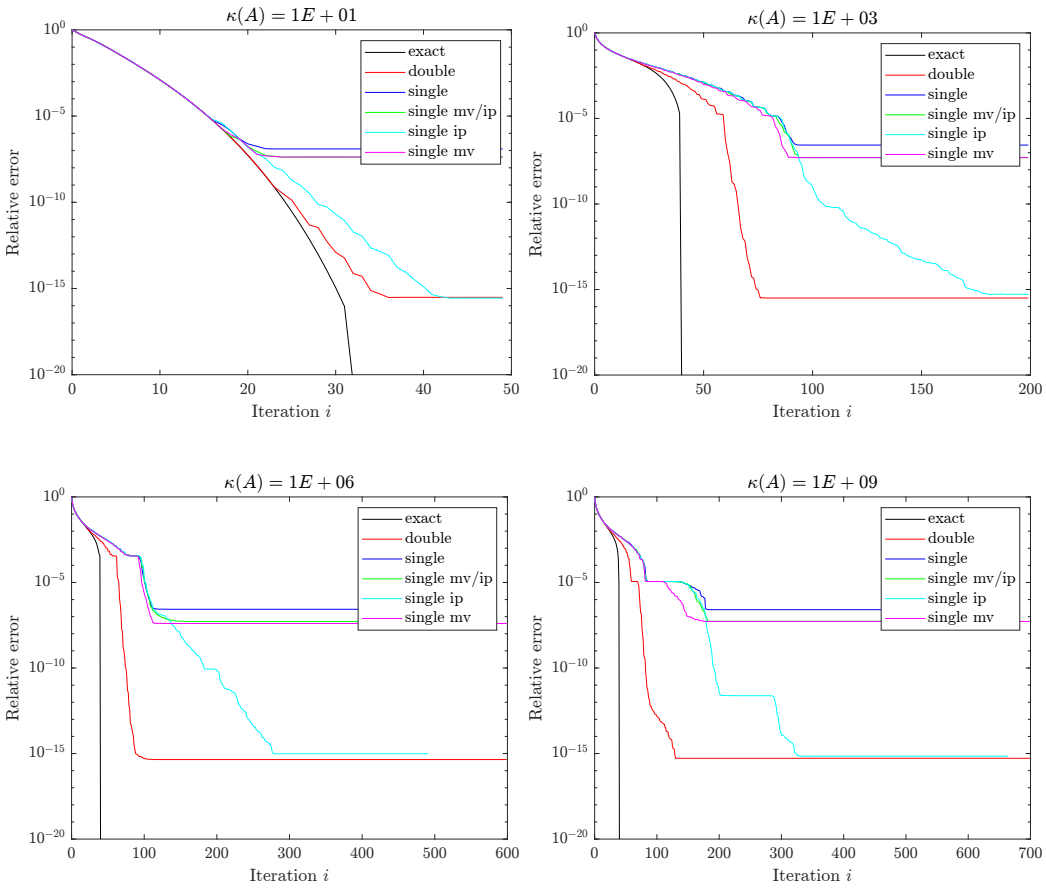

Figure 14: Problems with set solution, $\rho = .9$

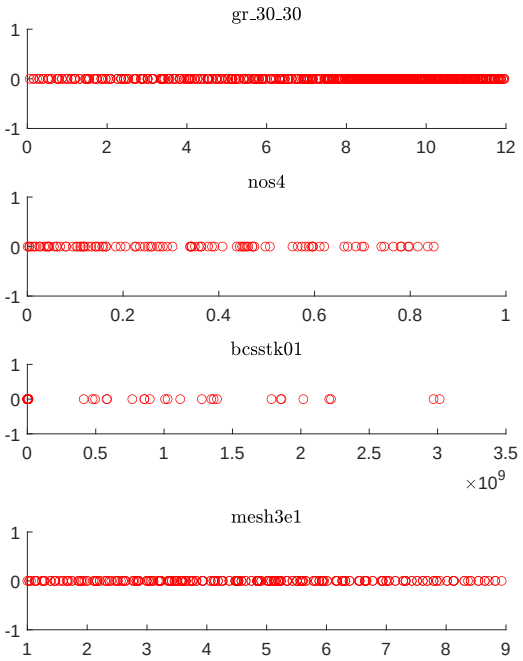

Figure 15: Spectra of Suitesparse Matrices.

2.4 A note on the use of selective higher precision

In addition to the described experiments, we also performed numerical experiments on this class of diagonal matrices where the working precision is lower and select computations are performed in higher precision. Unfortunately, in this case, it seems that there is no benefit to the use of mixed precision; those algorithms that used higher precision in either the SpMVs or the inner products behaved no differently numerically than the algorithm run entirely in the lower working precision. We believe that this can be explained theoretically.

2.5 Matrices from SuiteSparse

In addition to the diagonal test matrices, we have also tested the mixed precision approaches on many problems from the SuiteSparse collection [1]. We present a few of the results here. In Figure 15 we plot the spectra of the test matrices. For all problems, we generate the right-hand side b as in Section 2.1 (again, meant to present a difficult problem for CG) and use an initial guess of zero. The resulting convergence curves are shown in Figure 16.

The matrices gr_30_30 and mesh3e1 both have well-spaced eigenvalues across the whole spectra. From this we expect to see rather linear convergence behavior of exact CG, and this is indeed what we observe. This is especially pronounced for mesh3e1. The finite precision variants, both fixed double and single as well as mixed precision, all follow the exact CG convergence curve closely. In gr_30_30, 'single ip' only requires about 10% more iterations than CG in double precision and attains the same accuracy. For the problem mesh3e1, 'single ip' behaves exactly the same, in terms of both accuracy and convergence, as CG in double precision.

For the matrix nos4, the convergence of double precision CG is not too different from exact CG. The same is true of the variants which selectively use single precision up until a relative accuracy of around 10^{-8}. At this point, all variants that use single precision matrix-vector products stagnate. The 'single ip' variant continues converging, albeit with a significantly slower rate than exact or double CG after this point.

The problem bcsstk01 is a different story. From looking at the spectrum, we can see that there are large outlying eigenvalues. It is known that such a situation can cause convergence delays in finite precision CG; the underlying Lanczos process will create multiple Ritz values approximating the large eigenvalues of A. Indeed, we can see that this is a difficult problem for CG even entirely in double precision. These effects are amplified when single precision is used within the computation, resulting in even further delay of convergence.

3 Exploration of Convergence Behavior

We seek to understand the finite precision convergence behavior of mixed precision variants. Actually, based on our numerical experiments, it seems that the convergence behavior is similar among all variants that use some

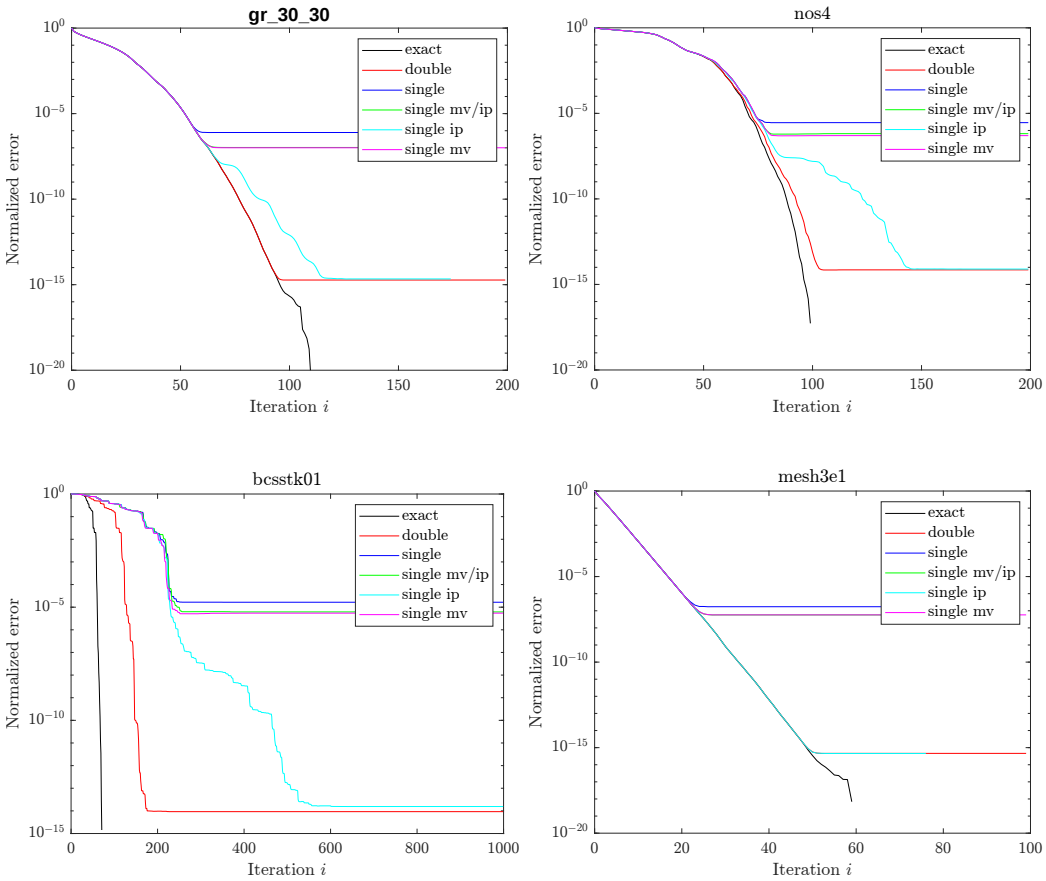

Figure 16: Matrices from the SuiteSparse Collection

form of lower precision; e.g., the convergence behavior (before attainable accuracy is reached) of CG entirely in single precision is similar to that of CG with single precision inner products and everything else in double.

Anne Greenbaum showed in her seminal work [2], based on the work in [6], that finite precision CG on a matrix A behaves like exact CG on a larger matrix $\tilde{A}$ which has eigenvalues in tight clusters around the eigenvalues of A. The bounds derived on the size of these clusters is on the order of $\sqrt{\varepsilon}\|A\|$, although this is not a tight bound. It is expected that one can prove a tighter bound is on the order of $\varepsilon\|A\|$. This is what is observed experimentally, but despite significant efforts of many in the field, a proof remains elusive.

We conjecture that the following can be observed: for mixed precision variants, mixed precision CG on A behaves like exact CG on a larger matrix $\tilde{A}$ which has eigenvalues in clusters around the eigenvalues of A, where these clusters have diameter on the order of $\varepsilon'\|A\|$, where $\varepsilon' = \max\{\varepsilon, \varepsilon_{IP}, \varepsilon_{MV}\}$. We perform an experiment to demonstrate this.

We generate a matrix A using the formula in (2.1) with $N = 25$, $\lambda_1 = 0.1$, $\lambda_N = 100$, and $\rho = 0.65$. We then create two matrices $\tilde{A}_1$ and $\tilde{A}_2$ by splitting each eigenvalue of A into clusters. In $\tilde{A}_1$, each cluster has 4 eigenvalues, and clusters have a diameter on the order 10^{-14}. In $\tilde{A}_2$, each cluster has 10 eigenvalues, and clusters have a diameter on the order 10^{-7}. In Figure 17, we run exact CG on $\tilde{A}_1$ and $\tilde{A}_2$, double precision CG on A, and the 'single ip' variant of CG on A, in which the working precision is double and inner products are computed in single precision. The right-hand side for each problem was set such that $\|b\|_2 = 1$ with identical entries. Similar experiments have been performed for fixed (double) precision CG; see, e.g., [5, Figure 5.16].

As we can see, just as Greenbaum's theory predicts that double precision CG on A will behave like exact CG on $\tilde{A}_1$, we can draw a similar conclusion about CG with single precision inner products. CG with single precision inner products behaves qualitatively like exact CG on a matrix whose eigenvalues are clustered around the eigenvalues of A, but the clusters have greater diameter and a greater number of eigenvalues per cluster than in the double precision case. It is clear that similar conclusions can be drawn for cases where 16-bit arithmetic is used for the inner products, but in this case the clusters will have even greater diameter (proportional to the unit roundoff).

Future theoretical investigation will involve proving what we observe here by extending the analyses of Paige [6] and Greenbaum [2]. However, as in the fixed precision case, we expect that we will obtain a looser bound than what is observable in practice, namely, clusters with diameter on the order of $\sqrt{\varepsilon'}\|A\|$. Overcoming this and proving a tighter bound would be of great interest not only for mixed precision variants, but for fixed precision variants as well.

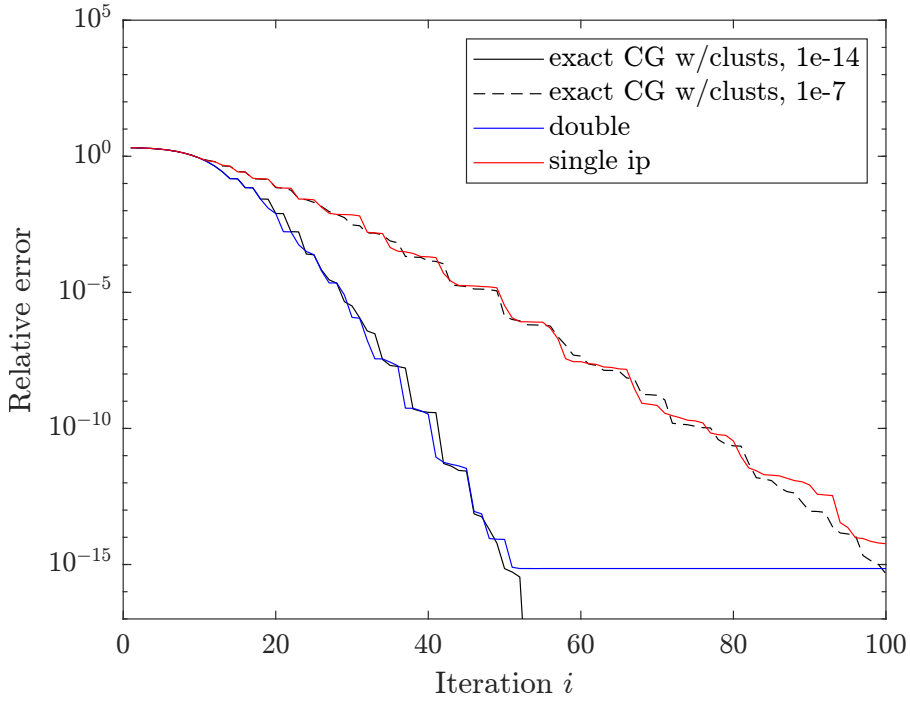

Figure 17: Comparison of exact CG on matrices with clusters with both double precision CG and double precision CG with single precision inner products

4 Conclusions and Further Outlook

In summary, we believe there is potential for the use of mixed precision within CG. Our experiments and analysis sketch show that inner products within CG can be computed at half of the working precision without significantly affecting the attainable accuracy. However, in order to be beneficial, we need a scenario in which the selective use of lower precision is both beneficial to performance and does not significantly affect the rate of convergence so as to negate this performance benefit. The study of convergence behavior is thus crucial; we need to identify properties of the matrix and right-hand side that indicate that the convergence will not deteriorate significantly when low precision is used. Our experiments in Section 2.2 illustrate the importance of taking the right-hand side into account. A thorough study of the potential performance of low-precision variants will need to involve collaboration with others involved in the xSDK project.

Further, it is clear from our experiments that the convergence behavior is similar for *all* variants that use some lower precision in some (or all) of the computation. The only difference is the attainable accuracy. If only low accuracy is needed, then it may be more beneficial to do all computations in the lower precision. However, if high accuracy is needed, then a mixed precision approach may be attractive. One hope is to be able to determine the right approach to take automatically; given a matrix, a right-hand side, and a desired accuracy, we would like to be able to determine whether to use a high precision, low precision, or mixed precision approach.

References

[1] Timothy A Davis and Yifan Hu. The University of Florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1–25, 2011.

[2] A. Greenbaum. Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences. *Lin. Alg. Appl.*, 113:7–63, 1989.

[3] A. Greenbaum. Estimating the attainable accuracy of recursively computed residual methods. *SIAM J. Matrix Anal. Appl.*, 18(3):535–551, 1997.

[4] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436, 1952.

[5] J. Liesen and Z. Strakoš. *Krylov subspace methods: principles and analysis.* Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, 2013.

[6] C. C. Paige. Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem. *Lin. Alg. Appl.*, 34:235–258, 1980.

[7] Henk A Van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on scientific and Statistical Computing*, 13(2):631–644, 1992.