Received: 3 January 2019 | Revised: 18 December 2019 | Accepted: 11 May 2020

DOI: 10.1002/nla.2315

RESEARCH ARTICLE

WILEY

Preconditioning for accurate solutions of ill-conditioned linear systems

Qiang Ye

Department of Mathematics, University of Kentucky, Lexington, Kentucky,

Correspondence
Qiang Ye, Department of Mathematics, University of Kentucky, Lexington, KY 40506.
Email: qye3@uky.edu

Funding information
NSF Division of Mathematical Sciences, Grant/Award Numbers: 1318633, 1620082, 1821144

Summary

This article develops the preconditioning technique as a method to address the accuracy issue caused by ill-conditioning. Given a preconditioner M for an ill-conditioned linear system $Ax = b$, we show that, if the inverse of the preconditioner M^{-1} can be applied to vectors *accurately*, then the linear system can be solved *accurately*. A stability concept called *inverse-equivalent* accuracy is introduced to describe the high accuracy that is achieved and an error analysis will be presented. Numerical examples are presented to illustrate the error analysis and the performance of the methods.

KEYWORDS

accuracy, error analysis, ill-conditioned linear systems, preconditioning

1 | INTRODUCTION

Solutions of large-scale linear algebra problems such as those arising from discretization of PDEs are often associated with an ill-conditioned matrix $A \in \mathbb{R}^{n\times n}$, where the condition number $\kappa(A) := \|A\|\|A^{-1}\|$ is large. The ill-conditioning has two effects in numerically solving a linear system $Ax = b$. It usually reduces the rate of convergence of iterative algorithms such as the Krylov subspace methods. It also limits the accuracy to which $Ax = b$ can be solved in finite precision. The former problem is typically addressed by a technique known as preconditioning. For the latter, there is no known good solution other than the classical diagonal scaling (or equilibration) or iterative refinements.[1(sec2.5),2(p124)]

The preconditioning technique is a general methodology that has been highly successful in overcoming the effect of ill-conditioning on the speed of convergence of iterative methods for solving a linear system $Ax = b$.[3] Given an invertible $M \approx A$, the preconditioning method implicitly transforms the linear system to the well-conditioned one, $M^{-1}Ax = M^{-1}b$, which can be solved iteratively with accelerated convergence. This poses a natural question: Do we also obtain a more accurate solution by solving the preconditioned system $M^{-1}Ax = M^{-1}b$? The answer is generally no. This is because, for M to be a good preconditioner to an ill-conditioned A, it is necessarily ill-conditioned and hence there are potentially large roundoff errors encountered in forming the preconditioned system either explicitly or implicitly, assuming a backward stable algorithm is used to solve preconditioning systems $Mu = v$; see Sections 3 and 4 for more details and examples. On the other hand, if the system $M^{-1}Ax = M^{-1}b$ can be formed exactly or sufficiently accurately (ie, $M^{-1}A$ and $M^{-1}b$ are computed with small relative errors in norm), solving the resulting well-conditioned system will give an accurate solution. Indeed, diagonal scaling is such an example where M is chosen to be a diagonal matrix of powers of 2 so that no roundoff error is generated when applying M^{-1}. This gives rise to the main idea of this article that solving preconditioning system $Mu = v$ more accurately than the standard backward stability in preconditioning may lead to improved solution accuracy.

We will develop the preconditioning technique as a method to solve the accuracy issue caused by ill-conditioning. We will show that preconditioning may indeed produce highly satisfactory solution accuracy of a linear system if the inverse of the preconditioner, M^{-1}, can be applied sufficiently *accurately*. To study precisely the accuracy that is needed for the application of M^{-1} and that can be attained by the final solution, we will introduce a stability concept called *inverse-equivalent* accuracy, which is one numerically equivalent to multiplying the right-hand side by exact inverses. An error analysis together with numerical examples will be presented to demonstrate that an appropriate formulation of the preconditioning method can produce solutions with *inverse-equivalent* accuracy if the preconditioning equation $Mu = v$ can be solved with *inverse-equivalent* accuracy. This condition on the preconditioner can be satisfied if M^{-1} is accurately available or if M has an *accurate rank-revealing decomposition* (see References 4-6 or Section 3.3).

We note that iterative refinement is a classical technique of using higher precision to deal with ill-conditioning[1(p60)]. Recently, Carson and Higham[7,8] have combined preconditioned iterative methods with the iterative refinement to solve an ill-conditioned system. Namely, they consider factorizing the coefficient matrix A only very approximately in a lower precision (eg, half-working precision) and use it as a preconditioner to the correction equation in the iterative refinement. It is shown that if the matrix-vector product for the preconditioned correction equation $M^{-1}Av$ is computed in a higher precision (eg, double-working precision), then the iterative refinement will lead to a solution with full accuracy. Here, we also explore the idea of more accurate applications of preconditioning but use special preconditioners with a fixed precision throughout.

The rest of the article is organized as follows. We present in Section 2 the concept of *inverse-equivalent* accuracy. We then develop and analyze in Section 3 the preconditioning method for improving solution accuracy. Finally, in Section 4, we present some numerical examples, followed by some concluding remarks in Section 5.

1.1 | Notation and preliminaries

Throughout this article, inequalities and absolute value involving matrices and vectors are entrywise. $\|\cdot\|$ denotes a vector norm on $\mathbb{R}^n$ as well as its induced matrix operator norm on $\mathbb{R}^{n\times n}$ that satisfies $\|Z\| = \||Z|\|$ and $\|Y\| \le \|Z\|$ if $|Y| \le |Z|$ where Y, Z are any $n \times n$ matrices or any $n \times 1$ vectors. This is the case for the 1-norm $\|\cdot\|_1$ and the ∞-norm $\|\cdot\|_\infty$, for example. We note that only some of our proofs use these special properties of norm; most of our proofs are valid for a general norm.

For error analysis in a floating point arithmetic, $\mathbf{u}$ denotes the machine roundoff unit and $\mathcal{O}(\mathbf{u})$ denotes a term bounded by $p(n)\mathbf{u}$ for some polynomial $p(n)$ in n. We use $fl(z)$ to denote the computed result of an algebraic expression z. We assume throughout that matrices and vectors given have floating point number entries. We assume the following standard model for roundoff errors in matrix computations[2(p66)]:

$$fl(x + y) = x + y + e \quad \text{with } |e| \le \mathbf{u}(|x + y|) \tag{1}$$

$$fl(Ax) = Ax + e \quad \text{with } |e| \le \mathbf{u}N|A||x| + \mathcal{O}(\mathbf{u}^2), \tag{2}$$

where N is the maximal number of nonzero entries per row of A. Using (2.4.12) of Reference 2, p64, and equivalence of any two norms in a finite dimensional space, we may also simply rewrite (2) as

$$\||fl(Ax) - Ax\| \le \mathcal{O}(\mathbf{u})N\|A\|\|x\|. \tag{3}$$

This bound is based on explicitly multiplying A with x and $N \le n$ can be absorbed into the $\mathcal{O}(\mathbf{u})$ term. More generally, if A is not explicitly given and Ax is computed as an operator, (3) may still be valid if we allow N to be a suitable constant associated with the operator Ax.

2 | INVERSE-EQUIVALENT ACCURACY

In this section, we introduce a stability concept called *inverse-equivalent* accuracy for solving linear systems in finite precision.

Given an invertible matrix $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$, standard dense algorithms for solving the linear system $Ax = b$ in a floating point arithmetic computes a solution $\hat{x}$ that is normwise backward stable, that is, it satisfies $(A + E)\hat{x} = b$ for some E with $\|E\|/\|A\| = \mathcal{O}(\mathbf{u})$. An iterative method computes a solution $\hat{x}$ with a residual that at best satisfies $\|b - A\hat{x}\| = \mathcal{O}(\mathbf{u})\|A\|\|\hat{x}\|$ at convergence. In both cases, the solution error is bounded as

$$\frac{\|\hat{x} - x\|}{\|x\|} \leq \mathcal{O}(\mathbf{u})\kappa(A), \quad \text{where } \kappa(A) = \|A\|\|A^{-1}\|. \tag{4}$$

This bound implies large solution errors for ill-conditioned problems, but for a general linear system, this is expected because the solution is sensitive under general normwise perturbations. If we consider entrywise small perturbations to A and b, then the perturbation to the solution is determined by the Bauer-Skeel condition number $\text{cond}(A) := \| \, |A^{-1}||A| \, \|$,[9] which is invariant under a row scaling of A and is much smaller than $\kappa(A)$ if the rows of A are badly scaled. Indeed, it has been shown that a solution accuracy with $\kappa(A)$ in (4) replaced by $\text{cond}(A)$ can be achieved by a variation of the Gaussian elimination,[9] the Cholesky factorization,[10] or equilibration.[2(p124) or 11(p177)] Thus, we are here interested in ill-conditioning that is not due to bad scaling.

To improve the accuracy (4), one may ideally like the full relative accuracy

$$\frac{\|\hat{x} - x\|}{\|x\|} \leq \mathcal{O}(\mathbf{u}) \tag{5}$$

but this being totally independent of A will obviously require very stringent conditions on A, as a perturbation to b alone will produce errors proportional to A^{-1}. With this in mind, we define below an accuracy measure that has been studied in various contexts.[5,11,12]

Definition 1. Given A, we say that an algorithm for solving linear systems with coefficient A is inverse-equivalent if, for any b, it produces in a floating point arithmetic a computed solution $\hat{x}$ to $Ax = b$ such that

$$\|\hat{x} - x\| \leq \mathcal{O}(\mathbf{u})\|A^{-1}\|\|b\|. \tag{6}$$

We also say such a solution $\hat{x}$ has inverse-equivalent accuracy. Furthermore, we say that the algorithm is strongly inverse-equivalent and $\hat{x}$ has strongly inverse-equivalent accuracy if

$$\|\hat{x} - x\| \leq \mathcal{O}(\mathbf{u})\| \, |A^{-1}||b| \, \|. \tag{7}$$

Clearly, strongly inverse-equivalent accuracy (7) implies inverse-equivalent accuracy (6). The next two results explain the naming of this accuracy.

Theorem 1. *If A is such that A^{-1} is explicitly available, then solving $Ax = b$ by multiplying A^{-1} with b is a strongly inverse-equivalent algorithm.*

Proof. Recall that A and b are assumed to have floating point number entries. For A^{-1}, we have $|fl(A^{-1}) - A^{-1}| \leq \mathbf{u}|A^{-1}|$. Then $|fl(A^{-1})b - A^{-1}b| \leq \mathbf{u}|A^{-1}||b|$. Recall that $fl(A^{-1}b)$ denotes the final computed result of $A^{-1}b$, that is, $fl(fl(A^{-1})b)$. It follows from (2) that $|fl(A^{-1}b) - fl(A^{-1})b| \leq \mathcal{O}(\mathbf{u})|fl(A^{-1})||b|$. Combining the two, we obtain

$$|fl(A^{-1}b) - A^{-1}b| \leq |fl(A^{-1}b) - fl(A^{-1})b| + |fl(A^{-1})b - A^{-1}b| \leq \mathcal{O}(\mathbf{u})|fl(A^{-1})||b|.$$

Thus $\|fl(A^{-1}b) - A^{-1}b\| \leq \mathcal{O}(\mathbf{u})\| \, |A^{-1}||b| \, \|$. ∎

Theorem 2. *Let A be an invertible matrix. There is an inverse-equivalent algorithm for solving $Ax = b$ if and only if the inverse A^{-1} can be computed by some algorithm with a relative error of order $\mathcal{O}(\mathbf{u})$, that is, the computed inverse $\hat{X}$ satisfies*

$$\|\hat{X} - A^{-1}\| \leq \mathcal{O}(\mathbf{u})\|A^{-1}\|. \tag{8}$$

Similarly, there is a strongly inverse-equivalent algorithm for solving $Ax = b$ if and only if $\hat{X}$ has a columnwise relative accuracy, that is, $\|\hat{x}_i - x_i\| \leq \mathcal{O}(\mathbf{u})\|x_i\|$ for all i, where $\hat{x}_i$ and x_i are the ith columns of $\hat{X}$ and A^{-1}, respectively.

Proof. First assume that there is an inverse-equivalent algorithm for A. Using this algorithm to compute the inverse A^{-1} by solving $AX = I$, let the computed inverse be $\hat{X} = [\hat{x}_1, \hat{x}_2, \dots, \hat{x}_n]$ and write $X = A^{-1} = [x_1, x_2, \dots, x_n]$. Then $\hat{x}_i$ has inverse-equivalent accuracy, that is, written in the 1-norm, $\|\hat{x}_i - x_i\|_1 \le \mathcal{O}(\mathbf{u})\|A^{-1}\|_1\|e_i\|_1 = \mathcal{O}(\mathbf{u})\|A^{-1}\|_1$. Thus $\|\hat{X} - X\|_1 = \max_i\|\hat{x}_i - x_i\|_1 \le \mathcal{O}(\mathbf{u})\|A^{-1}\|_1$. By equivalence of norms, (8) is proved.

Similarly, if there is a strongly inverse-equivalent algorithm for A, then $\|\hat{x}_i - x_i\| \le \mathcal{O}(\mathbf{u})\| \, |A^{-1}||e_i| \, \| = \mathcal{O}(\mathbf{u})\|x_i\|$.

Conversely, from the computed inverse $\hat{X}$, solving $Ax = b$ by computing $\hat{x} = fl(\hat{X}b)$, we have

$$\begin{aligned}\|\hat{x} - x\| &\le \|fl(\hat{X}b) - \hat{X}b\| + \|\hat{X}b - A^{-1}b\| \\ &\le \mathcal{O}(\mathbf{u})\| \, |\hat{X}||b| \, \| + \|(\hat{X} - A^{-1})b\| \\ &\le \mathcal{O}(\mathbf{u})(\| \, |A^{-1}||b| \, \| + \| \, |\hat{X} - A^{-1}||b| \, \|) + \| \, |\hat{X} - A^{-1}||b| \, \|. \end{aligned} \tag{9}$$

Now, if $\hat{X}$ satisfies (8), then,

$$\|\hat{x} - x\| \le \mathcal{O}(\mathbf{u})(\|A^{-1}\| + \mathcal{O}(\mathbf{u})\|A^{-1}\|)\|b\| + \mathcal{O}(\mathbf{u})\|A^{-1}\|\|b\| = \mathcal{O}(\mathbf{u})\|A^{-1}\|\|b\|$$

and hence $\hat{x}$ has inverse-equivalent accuracy. On the other hand, if $\|\hat{x}_i - x_i\| \le \mathcal{O}(\mathbf{u})\|x_i\|$ for all i, we have

$$\begin{aligned}\| \, |\hat{X} - A^{-1}||b| \, \|_1 &= \| \sum_{i=1}^{n} |\hat{x}_i - x_i||b_i| \, \|_1 = \sum_{i=1}^{n} \|\hat{x}_i - x_i\|_1|b_i| \\ &\le \mathcal{O}(\mathbf{u}) \sum_{i=1}^{n} \|x_i\|_1|b_i| = \mathcal{O}(\mathbf{u})\| \sum_{i=1}^{n} |x_i| \, |b_i|\|_1 \\ &= \mathcal{O}(\mathbf{u})\| \, |A^{-1}||b| \, \|_1, \end{aligned}$$

where $b = (b_i)$. Using this in (9), we have $\|\hat{x} - x\|_1 \le \mathcal{O}(\mathbf{u})\| \, |A^{-1}||b| \, \|_1$, that is, $\hat{x}$ has strongly inverse-equivalent accuracy. This completes the proof. ∎

The above shows that an inverse-equivalent algorithm produces solutions that are comparable with the one obtained by multiplying the exact inverse with the right-hand side vector b. So, this accuracy should be highly satisfactory in many applications.

If we rewrite (6) in the relative error form

$$\frac{\|\hat{x} - x\|}{\|x\|} \le \mathcal{O}(\mathbf{u})\frac{\|A^{-1}\|\|b\|}{\|x\|}, \tag{10}$$

then it is clear that this accuracy is between the full relative accuracy (5) and the backward stable solution accuracy (4) as $\|x\| \le \|A^{-1}\|\|b\| \le \|A^{-1}\|\|A\|\|x\|$. Note that the bound (10) has also appeared in the study of perturbation theory for $Ax = b$ when only the right-hand side vector b is perturbed.[11,12] It has been observed that the bound (10) may be substantially smaller than (4).[5,12] For example, this occurs as long as b has a significant projection on some left singular vector u_k of A corresponding to a singular value σ_k that is far less than the largest one. Namely, if $\sigma_1 \ge \sigma_2 \ge \dots \ge \sigma_n$ are the singular values of A, then for any fixed k, $\|x\|_2 = \|A^{-1}b\|_2 \ge |u_k^T b|/\sigma_k$. Hence

$$\frac{\|A^{-1}\|_2\|b\|_2}{\|x\|_2} \le \frac{\sigma_k}{\sigma_n}\frac{\|b\|_2}{|u_k^T b|} \ll \|A\|_2\|A^{-1}\|_2 \tag{11}$$

if for some k, we have

$$\frac{\sigma_k}{\cos\angle(b, u_k)} \ll \sigma_1. \tag{12}$$

See References 5,12 for some more detailed discussions. Indeed, if $\sigma_k \approx \sigma_n$ for some k, then $\|x\|_2 \ge (\sum_{i=k}^{n}(u_i^T b)^2)^{1/2}/\sigma_k$ and

$$\frac{\|A^{-1}\|_2\|b\|_2}{\|x\|_2} \le \frac{\sigma_k}{\sigma_n}\frac{1}{\left(\sum_{i=k}^{n}\cos^2\angle(b, u_i)\right)^{1/2}}, \tag{13}$$

which may be a moderate number, provided $\sum_{i=k}^{n} \cos^2 \angle(b, u_i)$ is not too small.

We remark that b, being the input data in a practical problem, is unlikely to be nearly orthogonal to all singular vectors corresponding to singular values σ_k that are near the smallest one σ_n. For example, if b is a random vector, (12) may be easily satisfied. So we may expect the inverse-equivalent accuracy (10) to be significantly better than the backward stable one (4) when b is the input data.

3 | ACCURATE SOLUTIONS FOR LINEAR SYSTEMS

In this section, we present a formulation of the preconditioning method for solving an ill-conditioned linear system $Ax = b$ where there is a preconditioner M so that $Mu = v$ can be solved with inverse-equivalent accuracy. We show that solving the preconditioned equation $M^{-1}Ax = M^{-1}b$ results in inverse-equivalent accuracy and we present our analysis in two subsections, one for direct methods and one for iterative ones. Here, we first briefly discuss why we need an inverse-equivalent algorithm rather than a standard backward stable algorithm for solving the preconditioning equation $Mu = v$.

We observe that for $M^{-1}A$ to be well-conditioned, M is necessarily ill-conditioned (ie, has a condition number comparable to A). This is because

$$\frac{\kappa(A)}{\kappa(M^{-1}A)} \le \kappa(M) \le \kappa(M^{-1}A)\kappa(A). \tag{14}$$

Then the application of M^{-1} on A and on b cannot be computed accurately by an existing backward stable algorithm. Indeed, the computed result of the right-hand side $M^{-1}b$ is $M^{-1}b + f$ with the error f bounded by $\|f\|/\|M^{-1}b\| \le \mathcal{O}(\mathbf{u})\kappa(M)$. Similarly, the computed result of $M^{-1}A$ is $M^{-1}A + E$ with $\|E\|/\|M^{-1}A\| \le \mathcal{O}(\mathbf{u})\kappa(M)$. Thus, the preconditioned system obtained is

$$(M^{-1}A + E)y = M^{-1}b + f, \tag{15}$$

and then even its exact solution y can only be bounded as

$$\frac{\|y - x\|}{\|x\|} \le \mathcal{O}(\mathbf{u})\kappa(M)\kappa(M^{-1}A), \tag{16}$$

which by (14) is approximately $\mathcal{O}(\mathbf{u})\kappa(A)$. We conclude that the computed solution to $M^{-1}Ax = M^{-1}b$, after accounting the errors of applying M^{-1} using a backward stable algorithm, has a relative error of order $\mathbf{u}\kappa(A)$. So, the solution accuracy cannot be improved; see numerical examples in Section 4.

Note that the discussion above is for a general M solved by a backward stable algorithm. If $M^{-1}b$ and $M^{-1}A$ can be computed with normwise relative accuracy, that is, $\|f\|/\|M^{-1}b\| \le \mathcal{O}(\mathbf{u})$ and $\|E\|/\|M^{-1}A\| \le \mathcal{O}(\mathbf{u})$, then the exact solution of (15) has an error $\frac{\|y-x\|}{\|x\|} \le \mathcal{O}(\mathbf{u})\kappa(M^{-1}A + E) \approx \mathcal{O}(\mathbf{u})$ and the computed solution of the well-conditioned system (15) has also small relative error. However, the assumptions on f and E are unrealistic in general. This leads us to the following questions: Can more accurately inverting M result in a more accurate solution of the original system, and if so, what accuracy is needed for M^{-1}? The rest of this section provides answers to these questions.

Let $A = M + K$ be a splitting of A and let M be such that there is an inverse-equivalent algorithm for solving any linear system with M as a coefficient matrix. Then using M as a preconditioner, we form the preconditioned system

$$Bx = c, \quad \text{where } B := I + M^{-1}K, \ c := M^{-1}b. \tag{17}$$

This system may be formed explicitly or implicitly depending on whether we solve it by a direct or an iterative method, respectively, but it is important that B or its product with vectors is formed in the way as given in (17). We will show that solving the well-conditioned system (17) by any backward stable algorithm leads to an inverse-equivalent accurate solution (6), provided $\|K\|\|x\|/\|b\|$ is a moderate number.

The following two subsections provide detailed analysis by considering solving (17) first using a direct method and then using an iterative one.

3.1 | Direct method for preconditioned systems

We consider forming (17) explicitly and then solving it by a backward stable direct method. In this regard, we first need to compute $M^{-1}K$ column by column by solving n linear systems. Assume that these linear systems are solved by an inverse-equivalent algorithm for M. Then, each column of the computed result of $M^{-1}K$ has inverse-equivalent accuracy. We denote the computed result as $\hat{Z}$ and it satisfies $\|\hat{Z} - M^{-1}K\| \le \mathcal{O}(\mathbf{u})\|M^{-1}\|\|K\|$. Furthermore, the coefficient matrix $B = I + M^{-1}K$ is computed as $fl(I + \hat{Z})$, which has an error term bounded by $\mathbf{u}(1 + \|\hat{Z}\|)$ by (1). Combining the two error terms together and denoting the final computed result $fl(I + \hat{Z})$ as $\hat{B}$, we can write the total error as

$$\hat{B} = I + M^{-1}K + E = B + E, \quad \text{with } \|E\| \le \mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|). \tag{18}$$

Similarly, the computed result of $c := M^{-1}b$, denoted by $\hat{c} := fl(M^{-1}b)$ satisfies

$$\|\hat{c} - c\| \le \mathcal{O}(\mathbf{u})\|M^{-1}\|\|b\|. \tag{19}$$

Theorem 3. *Let $A = M + K$ with A and M being invertible and let $Ax = b$. Assume that there is an inverse-equivalent algorithm for solving $Mu = v$ so that the computed results of $B := I + M^{-1}K$ and $c := M^{-1}b$, denoted by $\hat{B}$ and $\hat{c}$, respectively, satisfy (18) and (19). Let $\hat{x}$ be the computed solution to $\hat{B}x = \hat{c}$ by a backward stable algorithm so that $\hat{x}$ satisfies*

$$(\hat{B} + F)\hat{x} = \hat{c}, \quad \text{with } \frac{\|F\|}{\|\hat{B}\|} \le \mathcal{O}(\mathbf{u}). \tag{20}$$

Let $\delta := (\|E\| + \|F\|)\|B^{-1}\|$ and assume that $\delta < 1$. Then

$$\frac{\|\hat{x} - x\|}{\|A^{-1}\|\|b\|} \le \mathcal{O}(\mathbf{u})\frac{\kappa(B)}{1 - \delta}\left(4 + \frac{\|K\|\|x\|}{\|b\|}\right). \tag{21}$$

In particular, if $\|M^{-1}\|\|K\| < 1$, then

$$\frac{\|\hat{x} - x\|}{\|A^{-1}\|\|b\|} \le \frac{\mathcal{O}(\mathbf{u})}{(1 - \delta)(1 - \|M^{-1}\|\|K\|)}.$$

Proof. First, let $f = \hat{c} - c$ or $\hat{c} = c + f$. Then $\|f\| \le \mathcal{O}(\mathbf{u})\|M^{-1}\|\|b\|$. Let $\tilde{B} = \hat{B} + F = B + E + F$ and rewrite (20) as $\tilde{B}\hat{x} = c + f$. From $(\|E\| + \|F\|)\|B^{-1}\| < 1$, it follows that $\tilde{B}$ is invertible and

$$\|\tilde{B}^{-1}\| \le \frac{\|B^{-1}\|}{1 - (\|E\| + \|F\|)\|B^{-1}\|} = \frac{\|B^{-1}\|}{1 - \delta}. \tag{22}$$

We also have

$$\|M^{-1}\| = \|BA^{-1}\| \le \|B\|\|A^{-1}\| \tag{23}$$

and

$$\|B^{-1}\|\|\hat{B}\| \le \|B^{-1}\|(\|B\| + \|E\|) \le \|B^{-1}\|\|B\| + \delta \le 2\|B^{-1}\|\|B\|. \tag{24}$$

Furthermore, using

$$1 = \|I\| = \|B - M^{-1}K\| \le \|B\| + \|M^{-1}\|\|K\|, \tag{25}$$

we can bound (18) as

$$\|E\| \le \mathcal{O}(\mathbf{u})(\|B\| + 2\|M^{-1}\|\|K\|) = \mathcal{O}(\mathbf{u})(\|B\| + \|M^{-1}\|\|K\|), \tag{26}$$

where in the last equality we have combined the coefficient 2 of $\|M^{-1}\|\|K\|$ into $\mathcal{O}(\mathbf{u})$ (ie, $2\mathcal{O}(\mathbf{u}) = \mathcal{O}(\mathbf{u})$ with our notation). Now, clearly $Bx = c$ and then $\tilde{B}x = c + Ex + Fx$. Combining this with $\tilde{B}\hat{x} = c + f$, we have

$$\hat{x} - x = -\tilde{B}^{-1}Ex - \tilde{B}^{-1}Fx + \tilde{B}^{-1}f$$

and then

$$\begin{aligned}\|\hat{x}-x\| &\le \|\tilde{B}^{-1}\|(\|E\|\|x\| + \|F\|\|x\| + \|f\|)\\ &\le \frac{\|B^{-1}\|}{1-\delta}(\|E\|\|x\| + \|F\|\|x\| + \|f\|),\end{aligned} \tag{27}$$

where we have used (22). Further using (26), (20), (23), and (24), we have

$$\begin{aligned}\|\hat{x}-x\| &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\|B^{-1}\|\left((\|B\| + \|M^{-1}\|\|K\|)\|x\| + \|\hat{B}\|\|x\| + \|M^{-1}\|\|b\|\right)\\ &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}(\|B^{-1}\|\|B\|\|x\| + \|B^{-1}\|\|B\|\|A^{-1}\|\|K\|\|x\|\\ &\quad + 2\|B^{-1}\|\|B\|\|x\| + \|B^{-1}\|\|B\|\|A^{-1}\|\|b\|)\\ &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\|B^{-1}\|\|B\|\left(3\|x\| + \|A^{-1}\|\|K\|\|x\| + \|A^{-1}\|\|b\|\right)\\ &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\kappa(B)\left(4 + \frac{\|K\|\|x\|}{\|b\|}\right)\|A^{-1}\|\|b\|,\end{aligned} \tag{28}$$

where we have used $\|x\| \le \|A^{-1}\|\|b\|$ in the last inequality. This proves (21). Finally, if $\|M^{-1}\|\|K\| < 1$, then $B = I + M^{-1}K$ satisfies $\|B\| \le 1 + \|M^{-1}\|\|K\| \le 2$ and $\|B^{-1}\| \le \frac{1}{1-\|M^{-1}\|\|K\|}$. Applying these to (28), we have

$$\begin{aligned}\|\hat{x}-x\| &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\|B^{-1}\|(\|B\|\|x\| + \|x\| + 2\|B\|\|x\| + \|B\|\|A^{-1}\|\|b\|)\\ &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\frac{9}{1-\|M^{-1}\|\|K\|}\|A^{-1}\|\|b\|.\end{aligned}$$

Now the second bound follows from combining the factor 9 into the $\mathcal{O}(\mathbf{u})$ term. ∎

Note that $\delta = (\|E\| + \|F\|)\|B^{-1}\| \le \mathcal{O}(\mathbf{u})\|B^{-1}\|(1 + \|\hat{B}\| + \|M^{-1}\|\|K\|)$ can be expected to be much smaller than 1 and hence the factor $(1-\delta)^{-1}$ in the bounds is negligible. The second bound of the theorem shows that, when we have a very good preconditioner with $\|M^{-1}\|\|K\|$ bounded away from 1, then the inverse-equivalent accuracy is guaranteed. This is a rather strong condition as it implies $\|K\|/\|M\| \le 1/\kappa(M)$. However, when $\|M^{-1}\|\|K\| \ge 1$, the first bound (21) still holds, which implies that the inverse-equivalent accuracy of the solution may deteriorate by a factor of $\kappa(B)$ or $\frac{\|K\|\|x\|}{\|b\|}$. Such a dependence on $\kappa(B)$ and K is expected; however, as otherwise we would be able to solve any linear system $Ax = b$ more accurately by simply using $M = I$ as a preconditioner.

In the situation that M^{-1} is explicitly available, the computed result $fl(I + M^{-1}K)$ satisfies $|fl(I + M^{-1}K) - (I + M^{-1}K)| \le \mathcal{O}(\mathbf{u})(I + |M^{-1}||K|)$. Generally, if M has a strongly inverse-accurate algorithm, then the computed result of $B = I + M^{-1}K$, denoted by $\hat{B}$, has strongly inverse-equivalent accuracy:

$$\hat{B} = B + E, \quad \text{with } \|E\| \le \mathcal{O}(\mathbf{u})(1 + \|\,|M^{-1}||K|\,\|). \tag{29}$$

Similarly, the computed result of $M^{-1}b$, denoted by $\hat{c}$ satisfies

$$\hat{c} = c + f, \quad \text{with } \|f\| \le \mathcal{O}(\mathbf{u})\|\,|M^{-1}||b|\,\|. \tag{30}$$

In this case, we show that the computed solution $\hat{x}$ through preconditioning also has strongly inverse-equivalent accuracy.

Theorem 4. *Under the same assumption and notation of Theorem 3, if we assume additionally that the computed results $\hat{B}$ and $\hat{c}$, respectively, satisfy (29) and (30), then*

$$\frac{\|\hat{x}-x\|}{\|\,|A^{-1}||b|\,\|} \le \mathcal{O}(\mathbf{u})\frac{\kappa(B)}{1-\delta}\left(1 + \frac{\|\,|A^{-1}||K|\,\|\,\|x\|}{\|\,|A^{-1}||b|\,\|}\right). \tag{31}$$

Proof. We continue from the proof of Theorem 3. It follows from (29) and

$$1 = \|I\| = \|B - M^{-1}K\| \le \|B\| + \| \, |M^{-1}||K| \, \| \tag{32}$$

that

$$\|E\| \le \mathcal{O}(\mathbf{u})(\|B\| + 2\| \, |M^{-1}||K| \, \|) = \mathcal{O}(\mathbf{u})(\|B\| + \| \, |M^{-1}||K| \, \|).$$

Applying this, (20), and (30) to (27), and using (24) and $|M^{-1}| = |BA^{-1}| \le |B||A^{-1}|$, we have

$$\begin{aligned}
\|\hat{x} - x\| &\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\|B^{-1}\| \left(\|B\|\|x\| + \| \, |M^{-1}||K| \, \|\|x\| + \|\hat{B}\|\|x\| + \| \, |M^{-1}||b| \, \|\right) \\
&\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\|B^{-1}\| \left(\|B\|\|x\| + \| \, |B||A^{-1}||K| \, \|\|x\| + 2\|B\|\|x\| + \| \, |B||A^{-1}||b| \, \|\right) \\
&\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\|B^{-1}\|\|B\| \left(3\|x\| + \| \, |A^{-1}||K| \, \|\|x\| + \| \, |A^{-1}||b| \, \|\right) \\
&\le \frac{\mathcal{O}(\mathbf{u})}{1-\delta}\kappa(B) \left(4 + \frac{\| \, |A^{-1}||K| \, \| \, \|x\|}{\| \, |A^{-1}||b| \, \|}\right) \| \, |A^{-1}||b| \, \|,
\end{aligned}$$

where we have used $|x| \le |A^{-1}||b|$ in the last inequality. Combining four into $\mathcal{O}(\mathbf{u})$ term, the proof is complete. ∎

3.2 | Iterative method for preconditioned systems

For large-scale problems, we are more interested in solving the preconditioned system $Bx = c$ by an iterative method. In that case, the accuracy of the best approximate solution x_k obtained, as measured by the residual norm $\|Bx_k - c\|$, depends on the error made in computing matrix-vector product Bv. We first analyzed this error.

Lemma 1. *Let $B = I + M^{-1}K$ as in (17) and consider computing $Bv = v + M^{-1}Kv$ as in this expression for any $v \in \mathbb{R}^n$. If $M^{-1}Kv$ is computed by solving $Mw = Kv$ by an inverse-equivalent algorithm and if $fl(Bv)$ denotes the final computed result of Bv, then*

$$\|fl(Bv) - Bv\| \le \mathcal{O}(\mathbf{u})(1 + \|M^{-1}\| \, \|K\|)\|v\|. \tag{33}$$

If there is a strongly inverse-equivalent algorithm for inverting M, then

$$\|fl(Bv) - Bv\| \le \mathcal{O}(\mathbf{u})(1 + \| \, |M^{-1}| \, |K| \, \|)\|v\|. \tag{34}$$

Proof. First, if there is an inverse-equivalent algorithm for inverting M, let $u := Bv$ and denote the final computed result $fl(Bv)$ by $\hat{u}$. To compute Bv, we first compute Kv to get $fl(Kv) = Kv + e_1$ with $|e_1| \le n\mathbf{u}|K||v|$. Then computing $M^{-1}fl(Kv)$ by solving $Mw = fl(Kv)$ with the inverse-equivalent algorithm, the computed result, denoted by $\hat{w}$, satisfies

$$\begin{aligned}
\|\hat{w} - M^{-1}fl(Kv)\| &\le \mathcal{O}(\mathbf{u})\|M^{-1}\|\|fl(Kv)\| \\
&\le \mathcal{O}(\mathbf{u})\|M^{-1}\|(\|K\|\|v\| + \mathcal{O}(\mathbf{u})\|K\|\|v\|) \\
&= \mathcal{O}(\mathbf{u})\|M^{-1}\|\|K\|\|v\|.
\end{aligned}$$

Let $e_2 = \hat{w} - M^{-1}Kv$. Then

$$\begin{aligned}
\|e_2\| &= \|\hat{w} - M^{-1}fl(Kv) + M^{-1}e_1\| \\
&\le \mathcal{O}(\mathbf{u})\|M^{-1}\|\|K\|\|v\| + \mathcal{O}(\mathbf{u})\|M^{-1}\|\|K\|\|v\| \\
&= \mathcal{O}(\mathbf{u})\|M^{-1}\|\|K\|\|v\|.
\end{aligned}$$

Now, $\hat{u} = fl(v + \hat{w}) = v + \hat{w} + e_3$ with $|e_3| \le \mathbf{u}(|v| + |\hat{w}|)$. Then

$$\begin{aligned}\|e_3\| &\le \mathcal{O}(\mathbf{u})\|v\| + \mathcal{O}(\mathbf{u})(\|M^{-1}Kv\| + \|e_2\|)\\ &\le \mathcal{O}(\mathbf{u})(\|v\| + \|M^{-1}\|\|K\|\|v\| + \mathcal{O}(\mathbf{u})\|M^{-1}\|\|K\|\|v\|)\\ &= \mathcal{O}(\mathbf{u})(\|v\| + \|M^{-1}\|\|K\|\|v\|).\end{aligned}$$

Thus, we have $\hat{u} = v + M^{-1}Kv + e_2 + e_3 = u + e_2 + e_3$ and

$$\|\hat{u} - u\| \le \|e_2\| + \|e_3\| \le \mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|v\|.$$

On the other hand, if there is a strongly inverse-equivalent algorithm for inverting M, we modify the bounds for $\hat{w}$ and $M^{-1}e_1$ above as

$$\begin{aligned}\|\hat{w} - M^{-1}fl(Kv)\| &\le \mathcal{O}(\mathbf{u})\| \, |M^{-1}||fl(Kv)| \, \|\\ &\le \mathcal{O}(\mathbf{u})\| \, |M^{-1}|(|Kv| + \mathcal{O}(\mathbf{u})|K||v|) \, \|\\ &\le \mathcal{O}(\mathbf{u})\| \, |M^{-1}||K| \, \|\|v\|\end{aligned}$$

and $\|M^{-1}e_1\| \le n\mathbf{u}\| \, |M^{-1}||K||v| \, \| \le \mathcal{O}(\mathbf{u})\| \, |M^{-1}||K| \, \|\|v\|$. Using these in the rest of proof above, we obtain (34); the details are omitted. ∎

Now, apply an iterative method to the system $Bx = \hat{c}$, where $\hat{c} = fl(M^{-1}b)$. Since $B = I + M^{-1}K$ is assumed to be a small perturbation of I, the iteration is expected to converge quickly. However, the residual of the computed solution $\hat{x}_L$ obtained at iteration L can be at best $\mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|\hat{x}_L\|$, a level comparable with the error made in computing $B\hat{x}_L$. Note that most iterative methods update approximate solutions and the corresponding residuals at each iteration by general formulas of the forms $x_i = x_{i-1} + q_i$ (with $x_0 = 0$) and $r_i = r_{i-1} - Bq_i$ for some q_i. In a convergent iteration, the computed (or updated) residual $\hat{r}_i$ converges to 0, but the true residual $\hat{c} - B\hat{x}_i$ of the computed solution $\hat{x}_i$ deviates from $\hat{r}_i$ due to roundoff error accumulations and stagnates at certain level.[13-15] So at convergence when $\hat{r}_L$ is sufficiently small at some step L, $\hat{c} - B\hat{x}_L$ is primarily composed of the sum of roundoff errors made in computing matrix-vector product Bq_i for $i \le L$, which is bounded by $\mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\sum_{i=1}^{L}\|q_i\|$. Since $x_L = \sum_{i=1}^{L} q_i$, we may expect $\|\hat{x}_L\| \approx \sum_{i=1}^{L}\|q_i\|$ and hence $\hat{c} - B\hat{x}_L$ is approximately $\mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|\hat{x}_L\|$. Even if $\|\hat{x}_L\| \ll \sum_{i=1}^{L}\|q_i\|$, a residual replacement strategy[16,17] can be used to ensure $\hat{c} - B\hat{x}_i$ converges to the level $\mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|\hat{x}_L\|$.

Thus, we assume that the preconditioned system $Bx = \hat{c}$ is solved by an iterative method that produces $\hat{x}_L$ with residual approximately equal to $\mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|\hat{x}_L\|$. The next theorem demonstrates that this solution has an inverse-equivalent accuracy.

Theorem 5. *Consider solving (17) by an iterative method. Assume that the matrix-vector product $Bv = v + M^{-1}Kv$ is computed by solving $Mw = Kv$ by an inverse-equivalent algorithm and that the iterative method produces an approximate solution $\hat{x}_L$ with $\|\hat{c} - B\hat{x}_L\| \le \mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|\hat{x}_L\|$, where $\hat{c} = fl(M^{-1}b)$ satisfies (19). If $\|b - A\hat{x}_L\| \le \|b\|$, then*

$$\frac{\|x - \hat{x}_L\|}{\|A^{-1}\|\|b\|} \le \mathcal{O}(\mathbf{u})\kappa(B)\left(1 + \frac{\|K\|\|\hat{x}_L\|}{\|b\|}\right) \le \mathcal{O}(\mathbf{u})\kappa(B)\left(1 + \|A^{-1}\|\|K\|\right).$$

Proof. First we note that $\hat{x}_L = x - A^{-1}(b - A\hat{x}_L)$ and then

$$\|\hat{x}_L\| \le \|x\| + \|A^{-1}\|\|b - A\hat{x}_L\| \le \|x\| + \|A^{-1}\|\|b\| \le 2\|A^{-1}\|\|b\|.$$

As in the proof of Theorem 3, we have (25). Then it follows from (19) and the assumption on $\|\hat{c} - B\hat{x}_L\|$ that

$$\begin{aligned}\|c - B\hat{x}_L\| &\le \|\hat{c} - B\hat{x}_L\| + \|c - \hat{c}\|\\ &\le \mathcal{O}(\mathbf{u})(1 + \|M^{-1}\|\|K\|)\|\hat{x}_L\| + \mathcal{O}(\mathbf{u})\|M^{-1}\|\|b\|\\ &\le \mathcal{O}(\mathbf{u})(\|B\|\|\hat{x}_L\| + 2\|M^{-1}\|\|K\|\|\hat{x}_L\| + \|M^{-1}\|\|b\|).\end{aligned}$$

We now bound $x - \hat{x}_L = B^{-1}(c - B\hat{x}_L)$ as

$$\begin{aligned}\|x - \hat{x}_L\| &\le \mathcal{O}(\mathbf{u})\|B^{-1}\|(\|B\|\|\hat{x}_L\| + 2\|M^{-1}\|\|K\|\|\hat{x}_L\| + \|M^{-1}\|\|b\|)\\ &\le \mathcal{O}(\mathbf{u})\|B^{-1}\|(2\|B\|\|A^{-1}\|\|b\| + 2\|B\|\|A^{-1}\|\|K\|\|\hat{x}_L\| + \|B\|\|A^{-1}\|\|b\|)\\ &= \mathcal{O}(\mathbf{u})\kappa(B)\left(3 + 2\frac{\|K\|\|\hat{x}_L\|}{\|b\|}\right)\|A^{-1}\|\|b\|,\end{aligned}$$

where we have used (23). By combining the constants into the $\mathcal{O}(\mathbf{u})$ term, the first bound is proved. Bounding $\|\hat{x}_L\|$ by $2\|A^{-1}\|\|b\|$ again, and combining the factor 2 into $\mathcal{O}(\mathbf{u})$ term, we obtain the second bound of the theorem. ∎

Again, the solution obtained from the iterative method has inverse-equivalent accuracy provided $\kappa(B)$ and $\frac{\|K\|\|\hat{x}_L\|}{\|b\|}$ are not too large. A similar result for strongly inverse-equivalent accuracy is as follows.

Theorem 6. *Consider solving (17) by an iterative method. Assume that the matrix-vector product $Bv = v + M^{-1}Kv$ is computed by solving $Mw = Kv$ by a strongly inverse-equivalent algorithm and that the iterative method produces an approximate solution $\hat{x}_L$ with $\|\hat{c} - B\hat{x}_L\| \le \mathcal{O}(\mathbf{u})(1 + \| \, |M^{-1}| \, |K| \, \|)\|\hat{x}_L\|$, where $\hat{c} = fl(M^{-1}b)$ satisfies (30). If $|b - A\hat{x}_L| \le |b|$. Then*

$$\frac{\|x - \hat{x}_L\|}{\| \, |A^{-1}||b| \, \|} \le \mathcal{O}(\mathbf{u})\kappa(B)\left(1 + \| \, |A^{-1}||K| \, \|\right).$$

Proof. As in the proof of Theorem 5, it follows from $\hat{x}_L = x - A^{-1}(b - A\hat{x}_L)$ that

$$\|\hat{x}_L\| \le \|x\| + \| \, |A^{-1}||b - A\hat{x}_L| \, \| \le \|x\| + \| \, |A^{-1}||b| \, \| \le 2\| \, |A^{-1}||b| \, \|.$$

Clearly, (32) still holds. Then it follows from (30) and the assumption on $\|\hat{c} - B\hat{x}_L\|$ that

$$\begin{aligned}\|c - B\hat{x}_L\| &\le \mathcal{O}(\mathbf{u})(1 + \| \, |M^{-1}||K| \, \|)\|\hat{x}_L\| + \mathcal{O}(\mathbf{u})\| \, |M^{-1}||b| \, \|\\ &\le \mathcal{O}(\mathbf{u})(\|B\|\|\hat{x}_L\| + 2\| \, |M^{-1}||K| \, \|\|\hat{x}_L\| + \| \, |M^{-1}||b| \, \|).\end{aligned}$$

Now $x - \hat{x}_L = B^{-1}(c - B\hat{x}_L)$ is bounded as

$$\begin{aligned}\|x - \hat{x}_L\| &\le \mathcal{O}(\mathbf{u})\|B^{-1}\|(\|B\|\|\hat{x}_L\| + 2\| \, |M^{-1}||K| \, \|\|\hat{x}_L\| + \|B\|\| \, |A^{-1}||b| \, \|)\\ &\le \mathcal{O}(\mathbf{u})\|B^{-1}\|(2\|B\| + 4\|B\|\| \, |A^{-1}||K| \, \| + \|B\|)\| \, |A^{-1}||b| \, \|\\ &\le \mathcal{O}(\mathbf{u})\kappa(B)(3 + 4\| \, |A^{-1}||K| \, \|)\| \, |A^{-1}||b| \, \|.\end{aligned}$$

The theorem is proved by combining the constants into $\mathcal{O}(\mathbf{u})$. ∎

3.3 | Preconditioner

The key requirement to compute accurate solutions by preconditioning is that there is an inverse-equivalent algorithm for solving $Mu = v$. This is obviously the case if the inverse M^{-1} is available or can be accurately computed as in Theorem 2. More generally, if a preconditioner M has an *accurate* rank-revealing decomposition (RRD), then the solution to $Mx = b$ computed from the RRD is inverse-equivalent. The *accurate* rank-revealing decomposition is introduced by Demmel et al[4] to accurately compute the singular value decomposition of a matrix. Here is its definition for square matrices.

Definition 2 (4). A factorization $A = XDY$ of $A \in \mathbb{R}^{n\times n}$ is said to be rank-revealing if $X \in \mathbb{R}^{n\times n}$ and $Y \in \mathbb{R}^{n\times n}$ are well-conditioned and $D \in \mathbb{R}^{n\times n}$ is diagonal. Consider an algorithm for computing a rank-revealing decomposition $A = XDY$ and let $\hat{X}$, $\hat{D}$, and $\hat{Y}$ be the computed factors. We say $\hat{X}\hat{D}\hat{Y}$ is an *accurate rank-revealing decomposition* of A if $\hat{X}$ and $\hat{Y}$ are normwise accurate and $\hat{D}$ is entrywise accurate, that is,

$$\frac{\|\hat{X} - X\|}{\|X\|} \le \mathbf{u}p(n); \quad \frac{\|\hat{Y} - Y\|}{\|Y\|} \le \mathbf{u}p(n); \quad \text{and} \quad |\hat{D} - D| \le \mathbf{u}p(n)|D|, \tag{35}$$

where $p(n)$ is a polynomial in n.

As noted in Reference 4, the precise meaning of "well-conditioned" in the definition is not important as all related results involving this will be stated in terms of the condition numbers $\kappa(X)$ and $\kappa(Y)$, but in general, it refers to matrices with a condition number within a modest bound.

If A is invertible and $\hat{X}$, $\hat{D}$, and $\hat{Y}$ is the computed factors of an accurate rank-revealing decomposition of A , it is shown by Dopico and Molera[5(theorem 4.2)] that using it to solve $Ax = b$ through solving

$$\hat{X}y = b; \quad \hat{D}z = y; \quad \text{and} \quad \hat{Y}x = z.$$

with a backward stable algorithm for $\hat{X}s = b$ and $\hat{Y}x = w$, the computed solution $\hat{x}$ satisfies

$$\|\hat{x} - x\| \leq \mathcal{O}(\mathbf{u})\max\{\kappa(X), \kappa(Y)\}\|A^{-1}\|\|b\|.$$

Namely, $\hat{x}$ has inverse-equivalent accuracy provided $\kappa(X), \kappa(Y)$ are well-conditioned.

Several classes of matrices have been shown to have accurate RRD by Demmel et al,[4] which include graded matrices, total signed compound matrices such as acyclic matrices, Cauchy matrices, totally positive matrices, diagonally scaled totally unimodular matrices, and matrices arising in certain simple finite element problems. Diagonally dominant matrices have also been shown to have accurate rank-revealing decomposition.[6,18-20] Specifically, in Reference 6, algorithm 1, a variation of the Gaussian elimination is developed to compute an accurate *LDU* factorization from A and its diagonally dominant parts that is shown to be an accurate rank-revealing decomposition. The computational cost of this accurate *LDU* algorithm is about the same as the standard Gaussian elimination.

In applications, discretizations of differential operators are often diagonally dominant. When they are not, they may be close to being diagonally dominant, and then we can construct a diagonally dominant preconditioner, for which the accurate *LDU* factorization provides an inverse-equivalent algorithm. This will be used in our numerical examples in Section 4.

We remark that if two matrices A_1 and A_2 both have accurate rank-revealing decomposition, then solving $A_1A_2x = b$ through $A_1y = b$ and $A_2x = y$ will produce an inverse-equivalent solution provided $\|A_1^{-1}\|\ \|A_2^{-1}\|/\|(A_1A_2)^{-1}\|$ is a modest number.[21] In particular, we may also consider a preconditioner that is a product of diagonally dominant matrices; see Examples 2 and 4 in Section 4.

4 | NUMERICAL EXAMPLES

In this section, we present some numerical examples to demonstrate the accuracy achieved through preconditioning. All tests were carried out on a PC in MATLAB (R2016b) with a machine precision $\mathbf{u} \approx$ `2e-16`. The first two examples concern iterative solutions of large linear systems and the third example is on the direct method applied on the preconditioned equation for a small dense problem. The last two examples use two similar matrices as the first two examples with known exact eigenvalues to illustrate applications of inverse-equivalent accuracy to the eigenvalue computations.

For the first two examples, we consider linear systems arising in finite difference discretizations of some differential equations scaled so that the resulting matrix has integer entries. We construct an integer solution x so that $b = Ax$ can be computed exactly. Then x is the exact solution. In our testing, we are interested in systems with a random b, as this resembles practical situations, where b is usually the input data. By (12), a random b is also likely to yield a system where an inverse-equivalent accurate solution is significantly more accurate than a backward stable solution. To construct a random integer vector b with integer solution x, we first construct a random vector `b0=rand(n,1)` and set x_0 =`A\b0`, from which we construct a scaled integer solution x =`round(x0*1e8/norm(x0,inf))`, all in MATLAB functions. Then $b = Ax$ is computed exactly and is approximately a scaled random vector b_0.

The matrices in these examples do not possess any structure to allow computations of an accurate RRD. However, they are near a diagonally dominant matrix or a product of such, which is used as a preconditioner M. Then, computing the accurate *LDU* factorizations[6(algorithm1)] of the diagonal dominant matrices, the preconditioning equation $Mu = v$ is

solved with inverse-equivalent accuracy. We test the computed solutions $\hat{x}$ of the preconditioned equation $M^{-1}Ax = M^{-1}b$ with respect to the following errors:

$$\eta_{\text{ie}} := \frac{\|\hat{x} - x\|_2}{\|A^{-1}\|_2\|b\|_2} \quad \text{and} \quad \eta_{\text{rel}} := \frac{\|\hat{x} - x\|_2}{\|x\|_2}.$$

Here, η_{ie} measures the inverse-equivalent accuracy and η_{rel} is the relative accuracy. They differ by a fixed ratio $\frac{\eta_{\text{rel}}}{\eta_{\text{ie}}} = \frac{\|A^{-1}\|_2\|b\|_2}{\|x\|_2} \geq 1$. As a comparison, we also solve the preconditioning equation $Mu = v$ by the usual Cholesky factorization, which is only backward stable. This will test whether a standard preconditioning can result in any improvement in solution accuracy.

Example 1. Consider the 1-dimensional convection-diffusion equation

$$-u''(x) - u'(x) = f(x) \quad \text{on } (0, \gamma);$$

with the Dirichlet boundary condition $u(0) = u(\gamma) = 0$. Discretizing on a uniform grid of size $h = \gamma/(n+1)$ by the center difference scheme, we obtain $A_n = \frac{1}{h^2}T_n - \frac{1}{2h}K_n$, where T_n is the $n \times n$ tridiagonal matrix with diagonals being 2 and off-diagonals being -1, and K_n is the skew-symmetric $n \times n$ tridiagonal matrix with 1 on the superdiagonal above the main diagonal. To construct b and the exact solution $x = A_n^{-1}b$, we scale A_n by $2\gamma^2/(n+1)$ and use an integer value for γ so that the resulting matrix $2(n+1)T_n - \gamma K_n$ has integer entries. We then construct a random integer vector b and the corresponding exact solution x as discussed at the beginning of this section.

T_n is symmetric diagonally dominant and has an accurate LDL^T factorization. A_n is neither symmetric nor diagonally dominant but, if γ is not too large, preconditioning by $2(n+1)T_n$ yields a well-conditioned matrix $B = I - \frac{\gamma}{2(n+1)}T_n^{-1}K_n$. We solve the preconditioned equation $Bx = c$ by the GMRES method with the preconditioning equations $T_n u = v$ solved in two ways: (a) using the Cholesky factorization of T_n and (2) the accurate LDL^T factorization of T_n. The GMRES is implemented with restart after 50 iterations and the stopping tolerance for relative residual is set as $\sqrt{n}\mathbf{u}$. As a reference, we also solve the original system using MATLAB's division operator `An\b`. We compare the computed solutions $\hat{x}$ by the three methods with respect to η_{ie} and η_{rel}.

In Table 1, we present the results for a mildly ill-conditioned matrix with $n = 2^{13} - 1 = 8191$ and a more ill-conditioned one with $n = 2^{19} - 1 = 524287$. For each case of n, we test $\gamma = 10^1, 10^2, \ldots, 10^6$, resulting in an A_n that is increasingly not symmetric and not diagonally dominant. In the table, in addition to the errors η_{ie} and η_{rel}, we also present the condition numbers $\kappa_2(A_n)$ and, for the smaller n case, $\kappa_2(B)$ as well. (Computation of $\kappa_2(B)$ for the larger n case was beyond our computing resources.) In the columns for accurate LDU preconditioning, we also list $\rho := \|\gamma K_n\|_1\|x\|_1/\|b\|_1$, which is a factor in the error bound for η_{ie} by preconditioning; see Theorem 5.

We observe that the preconditioning produces an inverse-equivalent accuracy η_{ie} roughly in the order of machine precision regardless of the condition number $\kappa_2(A_n)$, when γ is a modest value (up to 10^4 in the first case and up to 10^2 in the second). These are the situation that T_n is still a good preconditioner. Taking into consideration the results of Example 2, η_{ie} appears to be proportional to $(1+\rho)\mathbf{u}$ as indicated by the theory. For the first case where $\kappa_2(B)$ is computed, η_{ie} increases slightly with $\kappa_2(B)$ but surprisingly, this effect appears to emerges only when $\kappa_2(B) \geq 10^5$. With η_{ie} in the order of machine precision, the relative error η_{rel} is improved accordingly, which, in this case, is near the machine precision. By contrast, the solutions by `A\b` and by the Cholesky preconditioning have relative errors η_{rel} of order $\kappa_2(A)\mathbf{u}$ as expected, which determines a corresponding η_{ie}. With larger γ, A_n becomes less ill-conditioned and the accuracy attained by `A\b` increases. When $\gamma \geq 10^4$ (the first n case) or $\gamma = 10^6$ (the second n case), it becomes more accurate than the one by the preconditioning, but since $\kappa_2(B)$ is larger than $\kappa_2(A_n)$ in those cases, the preconditioning is obviously not expected to be effective.

We have also computed the relative residual $\|b - A\hat{x}\|/(\|A\|\|\hat{x}\|)$, which measures the backward stability of the solution. It is always about 10^{-17} for `A\b`. It is also about 10^{-17} for the two preconditioning methods for γ up to 10^2 and then gradually increases to 10^{-14} as γ increases to 10^7. Namely, the solutions are also backward stable for at least modest values of γ.

Example 2. Let $A_n = (n+1)^4T_n^2 + \gamma S_n$, where T_n is as in Example 1, S_n is a random sparse integer matrix constructed using `Sn=floor(10*sprandn(n,n,0.001))` in MATLAB and γ is an integer parameter. Note that $(n+1)^4T_n^2$ is

TABLE 1 Example 1: accuracy for the three methods (A\b, Cholesky preconditioning, accurate LDL^T preconditioning): $\eta_{ie} = \|\hat{x} - x\|_2/(\|A^{-1}\|_2\|b\|_2)$, $\eta_{rel} = \|\hat{x} - x\|_2/\|x\|_2$, and $\rho = \|\gamma K_n\|_1\|x\|_1/\|b\|_1$

		A\b		Cholesky precond.			Accurate precond.		
γ	$\kappa_2(A_n)$	η_{ie}	η_{rel}	η_{ie}	η_{rel}	$\kappa_2(B)$	η_{ie}	η_{rel}	ρ
$n = 2^{13} - 1 = 8191$									
1e1	1e7	3e−12	4e−12	2e−12	3e−12	8e0	3e−15	4e−15	3e3
1e2	2e6	3e−11	4e−11	3e−13	3e−13	2e2	4e−15	5e−15	4e3
1e3	2e5	3e−12	4e−12	3e−14	4e−14	7e3	7e−15	9e−15	4e3
1e4	2e4	2e−17	3e−17	8e−15	1e−14	1e5	7e−15	9e−15	4e3
1e5	5e3	5e−16	6e−16	2e−14	3e−14	1e6	2e−14	3e−14	4e3
1e6	5e3	1e−15	2e−15	6e−13	8e−13	4e6	6e−13	8e−13	4e3
$n = 2^{19} - 1 = 524287$									
1e1	6e10	4e−10	5e−8	1e−9	2e−7	—	2e−16	2e−14	2e3
1e2	7e9	7e−9	1e−7	8e−10	2e−8	—	8e−15	2e−13	2e4
1e3	7e8	8e−9	2e−8	7e−10	2e−9	—	1e−13	4e−13	1e5
1e4	7e7	2e−9	2e−9	1e−10	2e−10	—	5e−14	6e−14	3e5
1e5	7e6	6e−11	8e−11	1e−11	2e−11	—	6e−14	8e−14	3e5
1e6	7e5	1e−18	2e−18	1e−12	2e−12	—	6e−14	8e−14	3e5

a finite difference discretization of 1-dimensional biharmonic operator $\frac{d^4u}{dx^4}$ with the boundary condition $u = \frac{d^2u}{dx^2} = 0$ on a uniform mesh on $[0, 1]$ with the meshsize $1/(n + 1)$. For an integer value of γ, A_n is an integer matrix and we construct a random integer vector b and the corresponding exact solution x as discussed at the beginning of this section.

If $|\gamma|$ is not too large, preconditioning with $(n + 1)^4T_n^2$ results in a well-conditioned matrix $B = I + \frac{\gamma}{(n+1)^4}T_n^{-2}S_n$. We solve the preconditioned equation $Bx = c$ by the GMRES method with the preconditioning equations $T_n^2u = v$ solved in two ways: (a) using the factorization $T_n^2 = R^TR$ computed from the QR factorization with column pivoting $T_n = QR$ and (b) using the accurate LDL^T factorization of T_n and solving T_n twice. The GMRES is implemented with restart after 50 iterations and the stopping tolerance for relative residual is set as $\sqrt{n}\mathbf{u}$. Again, we also solve the original system using MATLAB's division operator A\b. We compare the computed solutions $\hat{x}$ by the three methods with respect to η_{ie} and η_{rel}.

In Table 2, we present the testing results for $n = 2^{10} - 1 = 1023$ and $n = 2^{14} - 1 = 16383$. For these two cases, respectively, S_n has 1008 and 257 572 nonzeros with $\|S_n\|_\infty = 75$ and 343, respectively. For each of the n values, we test $\gamma = 10, -10^2, 10^3, -10^4, 10^5, -10^6, 10^7$. We list in the table $\kappa_2(A_n)$ and $\kappa_2(B)$ and $\rho := \|\gamma S_n\|_1\|x\|_1/\|b\|_1$, in addition to η_{ie}, η_{rel}.

We observe that the preconditioning produces an inverse-equivalent accuracy η_{ie} in the order of machine precision, except when $|\gamma|$ is as large as 10^6. Comparing with Example 1, η_{ie} is about 3 order of magnitude smaller and this seems to be due to a corresponding decrease in $1 + \rho$. As $|\gamma|$ increases, the quality of preconditioning deteriorates. However, as in Example 1, its effect on η_{ie} emerges only when $\kappa_2(B) \geq 10^5$. From that point on, η_{ie} appears proportional to $\kappa_2(B)(1 + \rho)\mathbf{u}$ as indicated by our theory. Computing the Cholesky factorization implicitly from the QR factorization with column pivoting of T_n has the beneficial effect in stability by avoiding forming T_n^2. This is similar to its advantage over the normal equation approach in the least squares problem. As a result, it significantly outperforms the backward stable solution A\b, where $A_n = (n + 1)^4T_n^2 + \gamma S_n$ is explicitly formed. The relative residual $\|b - A\hat{x}\|/(\|A\|\|\hat{x}\|)$ is always about 10^{-17} for A\b. For the two preconditioning methods, it gradually increases from 10^{-16} to 10^{-12} as γ increases to 10^7. We also note that $\frac{\eta_{rel}}{\eta_{ie}} = \frac{\|A^{-1}\|_2\|b\|_2}{\|x\|_2}$ is much larger for Example 2 than Example 1. This is likely due to a larger gap between the smallest eigenvalue and the other eigenvalues (ie, larger σ_k/σ_n in (13)) for the biharmonic operator. Other than this, similar behavior as in Example 1 is observed for this random sparse matrix.

The results of these two examples are in agreement with our error analysis (Theorem 5). The inverse-equivalent accuracy error η_{ie} appears proportional to $\kappa_2(B)(1 + \rho)\mathbf{u}$ although its dependence on $\kappa_2(B)$ may appear only when $\kappa_2(B)$ is quite

TABLE 2 Example 2: accuracy for the three methods (`A\b`, QR preconditioning, accurate LDL^T preconditioning): $\eta_{ie} = \|\hat{x} - x\|_2/(\|A^{-1}\|_2\|b\|_2)$, $\eta_{rel} = \|\hat{x} - x\|_2/\|x\|_2$, and $\rho = \|\gamma S_n\|_1\|x\|_1/\|b\|_1$

		`A\b`		**QR precond.**			**Accurate precond.**		
γ	$\kappa_2(A)$	η_{ie}	η_{rel}	η_{ie}	η_{rel}	$\kappa_2(B)$	η_{ie}	η_{rel}	ρ
$n = 2^{10} - 1 = 1023$									
1e1	2e11	3e−10	1e−7	8e−14	3e−11	3e0	5e−18	2e−15	2e−2
−1e2	2e11	8e−10	3e−7	8e−14	3e−11	9e1	6e−18	2e−15	2e−1
1e3	3e11	5e−11	3e−8	6e−14	3e−11	2e4	4e−18	2e−15	2e0
−1e4	4e10	2e−10	2e−8	3e−13	3e−11	2e5	6e−16	6e−14	1e1
1e5	9e9	2e−10	6e−9	5e−13	1e−11	5e6	7e−14	2e−12	1e2
−1e6	4e9	8e−11	1e−9	2e−10	2e−9	2e8	2e−11	3e−10	1e3
1e7	1e8	6e−13	2e−11	7e−11	2e−9	6e8	2e−11	6e−10	1e2
$n = 2^{14} - 1 = 16383$									
1e1	3e16	8e−10	5e−2	1e−16	7e−9	5e1	6e−19	3e−11	1e−6
−1e2	2e15	1e−10	4e−4	2e−16	6e−10	2e2	1e−18	3e−12	1e−5
1e3	6e14	2e−10	3e−4	5e−16	7e−10	9e3	6e−19	9e−13	1e−4
−1e4	5e14	5e−11	7e−5	3e−15	5e−9	8e5	5e−19	8e−13	9e−4
1e5	8e13	1e−10	3e−5	2e−14	4e−9	1e7	9e−17	2e−11	9e−3
−1e6	4e13	9e−11	1e−5	9e−13	1e−7	5e8	2e−15	2e−10	9e−2
1e7	9e12	3e−11	1e−6	1e−11	4e−7	1e10	8e−14	2e−9	9e−1

large. Indeed, its capability to produce a solution with inverse-equivalent accuracy with large $\kappa_2(B)$ is rather surprising. This would allow a broader application of the preconditioning method than what our theory might suggest.

Next, we give an example of small matrix where the inverse of the preconditioner is accurately available. Here, we test solving the preconditioned system by a direct method as discussed in Section 3.1 as well as the strongly inverse-equivalent accuracy.

Example 3. Let $H_n = [1/(i + j - 1)] \in \mathbb{R}^{n\times n}$ be the $n \times n$ Hilbert matrix. Its inverse $M_n = H_n^{-1}$ is an integer matrix that can be generated by MATLAB's `invhilb(n)`. This integer matrix is represented exactly in double precision for $n \le 14$. We consider a perturbation of the inverse Hilbert matrix: $A_n := M_n + S_n$, where S_n is a random dense integer matrix constructed using `Sn=floor(t*rand(n,n))` in MATLAB and t is an integer parameter. Then A_n is an integer matrix and we construct a linear system $A_n x = b$ with $x = [1, 1, \dots, 1]^T$ as the exact solution, that is, $b = A_n x$. This b vector can be represented exactly in double precision for $n \le 12$; so we use $n = 12$ in our experiment.

We consider solving this small dense problem by a direct method; see Section 3.1. We precondition A_n by M_n by explicitly forming the preconditioned system $Bx = c$, where $B = I + M_n^{-1}S_n$ and $c = M_n^{-1}b$; see (17). Then, we construct B and c in two ways: (1) standard preconditioning using MATLAB's division operator, that is, `B=eye(n)+Mn\Sn` and `c=Mn\b` and (b) preconditioning using the exact inverse of M_n, that is, `B=eye(n)+Hn* Sn` and `c=Hn* b`. Then, we solve the preconditioned system $Bx = c$ using MATLAB's division operator `B\c`. As a reference, we also present the results of solving $A_n v = b$ directly using MATLAB's division operator `An\b`.

Since we have an accurate inverse of M_n available, it follows from Theorem 4 that the solution by preconditioning with M_n has strongly inverse-equivalent accuracy. Therefore, we compare the computed solutions $\hat{x}$ in this example with respect to

$$\eta_{sie} := \frac{\|\hat{x} - x\|}{\| \, |A^{-1}||b| \, \|}$$

as well as the usual relative error η_{rel}. A^{-1} in the bound is computed using MATLAB's `vpa` with 400 digits. The computed inverses have residual norms near 10^{-400}. We also list η_{ie} for our method as a reference.

TABLE 3 Example 3: accuracy for the three methods (`A\b`, standard preconditioning `B=eye(n)+Hn* Sn`, accurate preconditioning `B=eye(n)+Hn* Sn`): $\eta_{\text{sie}} = \|\hat{x}-x\|_2 / \| \, |A^{-1}| \, |b| \, \|$, $\eta_{\text{ie}} = \|\hat{x}-x\|_2 / \left(\|A^{-1}\|_2 \| \, b| \, \|_2\right)$, $\eta_{\text{rel}} = \|\hat{x}-x\|_2/\|x\|_2$

			`A\b`		**Standard precond.**		**Accurate precond.**		
t	$\kappa_2(A_n)$	$\kappa_2(B)$	η_{sie}	η_{rel}	η_{sie}	η_{rel}	η_{sie}	η_{ie}	η_{rel}
1e0	1.7e16	1.0e0	1.2e−9	8.4e−2	1.2e−9	8.4e−2	2.8e−17	9.5e−18	2e−9
1e1	3.0e15	1.9e2	1.4e−10	1e−3	1.2e−9	9.2e−3	3.1e−16	6.5e−17	2.3e−9
1e2	5.3e14	5.0e3	5.2e−11	1.4e−4	1.2e−9	3.3e−3	6.8e−16	2.9e−16	1.9e−9
1e3	4.0e13	2.9e4	3.5e−10	4.7e−5	8.0e−10	1.1e−4	1.4e−14	3.9e−15	1.9e−9
1e4	2.4e14	2.7e7	8.5e−10	1.2e−3	9.2e−10	1.3e−3	4.4e−14	2.1e−14	6.3e−8
1e5	8.2e12	2.5e7	1.3e−10	4.6e−6	1.2e−9	4.2e−5	7.8e−13	2.8e−13	2.7e−8

In Table 3, we present the testing results for $t = 10^0, 10^1, 10^2, \ldots, 10^5$ and $n = 12$. We list in the table $\kappa_2(A_n)$ and $\kappa_2(B)$, in addition to η_{sie} and η_{rel}. The same behavior is observed as in Examples 1 and 2. The preconditioning approach results in strongly inverse-equivalent accuracy in the order of machine precision for $\kappa_2(B)$ up to 10^3, but as t and hence $\kappa_2(B)$ further increases, this accuracy is reduced proportionally, in agreement with Theorem 4. The accuracy of the other two methods are proportional to $\kappa_2(A_n)$, which decreases as t increases. For the standard preconditioning, this is shown in (16). Also $\frac{\eta_{\text{rel}}}{\eta_{\text{ie}}}$ is larger for this example compared with the previous two. As discussed before, this is likely due to the larger gap σ_k/σ_n for the inverse Hilbert matrix; see (13). Note that when $t = 1$, $S_n = 0$ and then the result in the first row of the table is basically the results of `Hn\b` and `Hn* b`. We have included this case to illustrate the difference in accuracy made by having the exact inverse.

Finally, we illustrate utility of the inverse-equivalent accuracy by considering accurate computation of the smallest eigenvalue (in absolute value) of a matrix through inverse iteration. It is well known that the computed smallest eigenvalue (in absolute value) of a symmetric matrix A by a backward stable algorithm generally has a relative error proportional to $\kappa_2(A)$. There have been significant body of works on theoretical and numerical studies of computing smaller eigenvalues accurately; see, for example, References 4,22-24 and the references contained therein. Noting that the largest eigenvalue (in absolute value) can be computed accurately independent of $\kappa_2(A)$, one way to compute the smallest eigenvalue accurately is by computing the largest eigenvalue of A^{-1} using the Lanczos algorithm/the power method. For this to succeed, we need to be able to compute the matrix-vector product $A^{-1}v$ accurately.[21] An inverse-equivalent algorithm for solving $Au = v$ would yield $A^{-1}v$ as accurately as the one obtained by multiplying the exact A^{-1} with v, which is sufficient to compute the largest eigenvalue of A^{-1} accurately. Thus, for a matrix A with a preconditioner M that has an inverse-equivalent algorithm for solving $Mu = v$, we can use a preconditioned iterative method to solve $Au = v$ with inverse-equivalent accuracy, which results in the largest eigenvalue of A^{-1} accurately. Its reciprocal, that is, the smallest eigenvalue of A, is then computed accurately. This approach works for both symmetric and nonsymmetric matrices.

Note that for many large-scale eigenvalue problems such as those arising in discretization of differential operators as in the examples below, the smallest eigenvalue is badly clustered but its reciprocal is typically very well separated after the inverse transformation.[25] Then all iterative methods such as the Lanczos algorithm or the inverse iteration converges quickly and produce similar results. Below, we report the results obtained by the inverse iteration only. In applying A^{-1} at each step of iteration, we solve $Au = v$ by a preconditioned iterative method. We test solving the preconditioner by the usual Cholesky factorization or by the accurate *LDU* factorization[6(algorithm1)]. With the two ways of solving the preconditioning equations, we test the accuracy in inverting A by comparing the final approximate eigenvalues obtained.

Example 4. Consider the eigenvalue problem for the 1-dimensional convection-diffusion operator as in Example 1: $-u''(x) - u'(x) = \lambda u(x)$ on $(0, \gamma)$ with $u(0) = u(\gamma) = 0$. The eigenvalues of this operator are exactly known[26(theorem1)]:

$$\lambda_i = \frac{1}{4} + \frac{\pi^2 i^2}{\gamma^2}, \quad \text{for } i = 1, 2, \ldots$$

Discretizing on a mesh of size $h = \gamma/(n+1)$ as in Example 1, we obtain the same matrix $A_n = \frac{1}{h^2} T_n - \frac{1}{2h} K_n$.

h	μ_1^{chol}	$\frac{\lvert\lambda_1-\mu_1^{\text{chol}}\rvert}{\lambda_1}$	μ_1^{aldu}	$\frac{\lvert\lambda_1-\mu_1^{\text{aldu}}\rvert}{\lambda_1}$
1.6e−2	10.11732544149765	2.3e−4	10.11732544149762	2.3e−4
3.9e−3	10.11946195350748	1.4e−5	10.11946195350759	1.4e−5
9.8e−4	10.11959549807000	8.8e−7	10.11959549806623	8.8e−7
2.4e−4	10.11960384467139	5.5e−8	10.11960384465017	5.5e−8
6.1e−5	10.11960436740018	3.3e−9	10.11960436631197	3.4e−9
1.5e−5	10.11960440025146	8.3e−11	10.11960439891543	2.1e−10
3.8e−6	10.11960357476229	8.2e−8	10.11960440095356	1.3e−11
9.5e−7	10.11959786966499	6.5e−7	10.11960440107954	9.7e−13
2.4e−7	10.11960179526253	2.6e−7	10.11960440108836	9.9e−14
6.0e−8	10.11996930306172	3.6e−5	10.11960440108905	3.0e−14

TABLE 4 Example 4: approximation of $\lambda_1 = \frac{1}{4} + \pi^2 = 10.11960440108936$ (μ_1^{chol}—computed eigenvalue by Cholesky preconditioner; μ_1^{aldu}—computed eigenvalue by accurate LDL^T preconditioner)

We approximate $\lambda_1 = \frac{1}{4} + \frac{\pi^2}{\gamma^2}$ by computing the smallest eigenvalue of A_n using the inverse iteration. At each iteration, we solve $A_n u = v$ by the GMRES method as preconditioned by $\frac{1}{h^2}T_n$ with two ways of solving equations involving the preconditioner T_n: 1. using the Cholesky factorization of T_n, and 2. the accurate LDL^T factorization of T_n. We denote the computed smallest eigenvalues by μ_1^{chol} and μ_1^{aldu}, respectively. The GMRES is implemented with restart after 50 iterations and the stopping tolerance for relative residual is set at $\sqrt{n}\mathbf{u}$. The stopping tolerance for the eigenvalue-eigenvector residuals of the inverse iteration is also set at $\sqrt{n}\mathbf{u}$. We use this very stringent criterion to ensure as accurate results as possible. In all cases, the inverse iteration terminates with the residual satisfying the criterion.

In Table 4, we present the testing results for $h = 2^{-6}, 2^{-8}, \ldots, 2^{-24}$ and $\gamma = 1$. We list the computed eigenvalues μ_1^{chol} and μ_1^{aldu} and their relative errors. As explained in Reference 21, the error $|\mu_1^{\text{chol}} - \lambda_1| \le |\lambda_{1,h} - \lambda_1| + |\mu_1^{\text{chol}} - \lambda_{1,h}|$ consists of the discretization error $\lambda_{1,h} - \lambda_1$ and the algebraic computational error $\mu_1^{\text{chol}} - \lambda_{1,h}$, where $\lambda_{1,h}$ is the (exact) smallest eigenvalue of A_n. As h decreases, the discretization error $\lambda_{1,h} - \lambda_1$ is known to converge to 0 quadratically in h, and we observe that μ_1^{chol} initially converge as expected. However, as h decreases, the matrix A_n becomes increasingly ill-conditioned and the roundoff errors associated with the standard Cholesky preconditioning increase. Then the algebraic error $\mu_1^{\text{chol}} - \lambda_{1,h}$ increases with $\kappa_2(A_n)$ from the level of the machine precision and will exceed the discretization errors at some point. In this example, this occurs at $h \approx 1.5\text{e}-5$, after which further decreasing h actually increases the error for μ_1^{chol}. On the other hand, preconditioning allows us to compute $A_n^{-1}v$ with an inverse-equivalent accuracy that is independent of the condition number $\kappa_2(A_n)$. Then $\mu_1^{\text{aldu}} - \lambda_{1,h}$ stays at the level of machine precision. Thus, the error for μ_1^{aldu} decreases quadratically to the order of machine precision. Note that the discretization matrix of the convection-diffusion operator is nonsymmetric and not diagonally dominant.

Example 5. Consider computing the smallest eigenvalue of the 1-dimensional biharmonic problem: $\frac{d^4v}{dx^4} + \rho v = \lambda v$ on $[0, 1]$ with the natural boundary condition $v(0) = v''(0) = v(1) = v''(1) = 0$. Discretizing on a uniform mesh of size $h = 1/(n+1)$ leads to $A_n = \frac{1}{h^4}T_n^2 + \rho I$, where T_n is the discretization of 1-dimensional Laplacian defined in Example 1. The eigenvalues of A_n are $\lambda_{j,h} = \frac{1}{h^4}16\sin^4(j\pi h/2) + \rho$[1(lemma6.1)]. We consider $n = 2^{15} - 1 = 32767$ for this example and $\rho = \pm\zeta, \pm 10\zeta, \pm 10^2\zeta, \pm 10^3\zeta$, where $\zeta = 0.5376671395461$ is a random number generated by MATLAB's `randn`. This results in an extremely ill-conditioned A_n with $\kappa_2(A_n) \approx 10^{18}$. A_n also becomes indefinite when $\rho = -10^3\zeta$.

We compute the smallest eigenvalue in absolute value, denoted by λ_{absmin}, of A_n by applying the inverse iteration to A_n. Note that this eigenvalue may not be $\lambda_{1,h}$ if A_n is indefinite. In carrying out the inverse iterations, we solve $A_n x = b$ by the CG (or MINRES if $\gamma < 0$) method as preconditioned by $\frac{1}{h^4}T_n^2$ with two ways of solving equations with the coefficient T_n^2: (a) using the factorization $T_n^2 = R^T R$ computed from the QR factorization with column pivoting $T_n = QR$ and (b) using accurate LDL^T factorization of T_n and solving equations with T_n twice. We denote the computed smallest eigenvalues in absolute value by μ_1^{qr} and μ_1^{aldu}, respectively. The stopping tolerance for relative residual of CG or MINRES is set at $\sqrt{n}\mathbf{u}$. The stopping tolerance for the eigenvalue-eigenvector residuals of the inverse iteration is also set at $\sqrt{n}\mathbf{u}$.

In Table 5, we present, for each case of ρ, the exact eigenvalue λ_{absmin}, the computed eigenvalues μ_1^{qr} and μ_1^{aldu} and their relative errors. For all the cases of ρ here, the preconditioned matrix $B = I + \rho h^2 T_n^{-2}$ is well conditioned with $\kappa(B)$ ranging between 1 and 7. As a result, the preconditioning approach produces μ_1^{aldu} that is accurate to the machine precision in

TABLE 5 Example 5: approximation of the smallest eigenvalue in absolute value λ_{absmin} (μ_1^{qr} and μ_1^{aldu}—computed eigenvalue by *QR* factorization and by accurate *LDU* factorization for the preconditioner, respectively, $e^{qr} := \frac{|\lambda_{\text{absmin}} - \mu_1^{qr}|}{|\lambda_{\text{absmin}}|}$; $e^{\text{aldu}} := \frac{|\lambda_{\text{absmin}} - \mu_1^{\text{aldu}}|}{|\lambda_{\text{absmin}}|}$)

ρ	λ_{absmin}	μ_1^{qr}	e^{qr}	μ_1^{aldu}	e^{aldu}
5e−1	97.946758024321	97.946756521067	2e−8	97.946758024319	3e−14
−5e−1	96.871423745229	96.871422234293	2e−8	96.871423745226	3e−14
5e0	102.785762280236	102.785760778849	1e−8	102.785762280234	2e−14
−5e0	92.032419489314	92.032417984375	2e−8	92.032419489312	2e−14
5e1	151.175804839385	151.175803328829	1e−8	151.175804839384	6e−14
−5e1	43.642376930165	43.642375423702	3e−8	43.642376930163	5e−14
5e2	635.076230430875	635.076228907167	2e−9	635.076230430879	6e−15
−5e2	−440.258048661325	−440.258050174851	3e−9	−440.258048661324	3e−15

all cases. The eigenvalues μ_1^{qr} computed using the preconditioning with the *QR* factorization have relative errors around 10^{-8}. Again we see that the preconditioning approach accurately computes the smallest eigenvalue of this extremely ill-conditioned matrix, even when the matrix is indefinite.

These two eigenvalue examples also confirm the higher accuracy achieved in computing $A^{-1}v$ by the preconditioning approach. They also illustrate that the inverse-equivalent accuracy in solving $Au = v$ leads to the largest eigenvalue of A^{-1} accurately.

5 | CONCLUDING REMARKS

We have shown that preconditioning with a preconditioner that can be solved with inverse-equivalent accuracy improves solution accuracy of an ill-conditioned linear systems. An error analysis is developed to demonstrate that the inverse-equivalent accuracy is achieved by this approach. Numerical examples confirm the analysis but also show that the method may work even when the quality of preconditioner is low. As an application, we use it to accurately compute the smallest eigenvalue of some differential operator discretizations that are indefinite or nonsymmetric.

For future works, it will be interesting to study a related perturbation theory and to investigate what sometimes appears to be a very mild dependence of the accuracy on the condition number of the preconditioned matrix. It will also be interesting to study whether our method can be used with preconditioners that are defined through their inverses, such as multilevel preconditioners[27] and sparse approximate inverse preconditioners.[28,29]

ACKNOWLEDGEMENTS

The author would like to thank Prof. Jinchao Xu for some interesting discussions on multilevel preconditioners that have inspired this work. The author would also like to thank Kasey Bray for many helpful comments on a draft of this article and also like to thank anonymous referees for many detailed and insightful comments that have significantly improved the paper. This research was supported in part by NSF under Grants DMS-1318633, DMS-1620082, and DMS-1821144. This work does not have any conflicts of interest.

ORCID

Qiang Ye https://orcid.org/0000-0002-8357-221X

REFERENCES

1. Demmel J. Applied numerical linear algebra. Philadelphia, PA: SIAM, 1997.
2. Golub G, Van Loan C. Matrix computations. 2nd ed. Baltimore, MD: The Johns Hopkins University Press, 1989.
3. Saad Y. Iterative methods for sparse linear systems. Philadelphia, PA: SIAM, 2003.
4. Demmel J, Gu M, Eisenstat S, Slapnicar I, Veselic K, Drmac Z. Computing the singular value decomposition with high relative accuracy. Linear Alg Appl. 1999;299:21–80.

5. Dopico FM, Molera JM. Accurate solution of structured linear systems via rank-revealing decompositions. IMA J Numer Anal. 2012;32:1096–1116.
6. Ye Q. Computing SVD of diagonally dominant matrices to high relative accuracy. Math Comp. 2008;77:2195–2230.
7. Carson E, Higham NJ. A new analysis of iterative refinement and its application to accurate solution of ill-conditioned sparse linear systems. SIAM J. Sci. Comp. 2017;39:A2834–A2856.
8. Carson E, Higham NJ. Accelerating the solution of linear systems by iterative refinement in three precisions. SIAM J Sci Comp. 2018;40:A817–A847.
9. Skeel R. Scaling for numerical stability in Gaussian elimination. J Assoc Comput Mach. 1979;26:494–526.
10. Demmel J. On floating point errors in Cholesky, LAPACK working note 14, UT-CS-89-87; October; 1989.
11. Higham NJ. Accuracy and stability of numerical algorithms. Philadelphia, PA: SIAM, 2002.
12. Chan T, Foulser D. Effectively well-conditioned linear systems. SIAM J Sci Stat Comput. 1988;9:963–969.
13. Greenbaum A. Estimating the attainable accuracy of recursively computed residual methods. SIAM J Matrix Anal Appl. 1997;18:535–551.
14. Gutknecht M. Lanczos-type solvers for nonsymmetric linear systems of equations. Acta Numerica. 1997;6:271–397.
15. Sleijpen GLG, van der Vorst HA, Fokkema DR. BICGSTAB(ℓ) and other hybrid Bi-CG methods. Numer Alg. 1994;7:75–109.
16. Sleijpen G, van der Vorst H. Reliable updated residuals in hybrid Bi-CG methods. Computing. 1996;56:144–163.
17. van der Vorst H, Ye Q. Residual replacement strategies for Krylov subspace iterative methods for the convergence of true residuals. SIAM J Sci Comput. 2000;22:836–852.
18. Alfa S, Xue J, Ye Q. Accurate computation of the smallest eigenvalue of a diagonally dominant M-matrix. Math Comput. 2002;71:217–236.
19. Demmel J, Koev P. Accurate SVDs of weakly diagonally dominant M-matrices. Numer Math. 2004;98:99–104.
20. Dopico FM, Koev P. Perturbation theory for the LDU factorization and accurate computations for diagonally dominant matrices. Numer Math. 2011;119:337–371.
21. Ye Q. Accurate inverses for computing eigenvalues of extremely ill-conditioned matrices and differential operators. Math Comp. 2018;87:237–259.
22. Barlow J, Demmel J. Computing accurate eigen systems of scaled diagonally dominant matrices. SIAM J Num Anal. 1990;27:762–791.
23. Demmel J, Kahan W. Accurate singular values of bidiagonal matrices. SIAM J Sci Stat Comput. 1990;11:873–912.
24. Li RC. Relative perturbation theory I: Eigenvalue and singular value variations. SIAM J Matrix Anal Appl. 1998;19:956–982.
25. Bai Z, Demmel J, Dongarra J, Ruhe A, van der Vorst H, editors. Templates for the solution of algebraic eigenvalue problems: A practical guide. Philadelphia, PA: SIAM, 2000.
26. Reddy S, Trefethen L. Pseudospectra of the convection-diffusion operator. SIAM J Appl Math. 1994;54:1634–1649.
27. Xu J. Iterative methods by space decomposition and subspace correction. SIAM Rev. 1992;34:581–613.
28. Benzi M, Meyer CD, Tuma M. A sparse approximate inverse preconditioner for the conjugate gradient method. SIAM J Sci Comp. 1996;17:1135–1149.
29. Benzi M, Tuma M. A sparse approximate inverse preconditioner for nonsymmetric linear systems. SIAM J Sci Comp. 1998;19:968–994.

How to cite this article: Ye Q. Preconditioning for accurate solutions of ill-conditioned linear systems. *Numer Linear Algebra Appl.* 2020;e2315. https://doi.org/10.1002/nla.2315