NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2008; **15**:489–513
Published online 7 January 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nla.570

Applications of statistical condition estimation to the solution of linear systems

A. J. Laub[1,2] and J. Xia[2,*,†]

[1] *Department of Electrical Engineering, University of California, Los Angeles, CA 90095, U.S.A.*
[2] *Department of Mathematics, University of California, Los Angeles, CA 90095, U.S.A.*

SUMMARY

This paper discusses some applications of statistical condition estimation (SCE) to the problem of solving linear systems. Specifically, triangular and bidiagonal matrices are studied in some detail as typical of structured matrices. Such a structure, when properly respected, leads to condition estimates that are much less conservative compared with traditional non-statistical methods of condition estimation. Some examples of linear systems and Sylvester equations are presented. Vandermonde and Cauchy matrices are also studied as representative of linear systems with large condition numbers that can nonetheless be solved accurately. SCE reflects this. Moreover, SCE when applied to solving very large linear systems by iterative solvers, including conjugate gradient and multigrid methods, performs equally well and various examples are given to illustrate the performance. SCE for solving large linear systems with direct methods, such as methods for semi-separable structures, are also investigated. In all cases, the advantages of using SCE are manifold: ease of use, efficiency, and reliability. Copyright © 2008 John Wiley & Sons, Ltd.

Received 23 May 2007; Revised 29 October 2007; Accepted 12 November 2007

KEY WORDS: linear systems; conditioning; statistical condition estimations

1. INTRODUCTION

The need to estimate the condition or sensitivity of numerical problems is one of the most fundamental issues in numerical analysis. Together with knowledge of the underlying stability of the algorithm employed to solve a problem, a good condition estimate can be used to provide a means to say something quantitative about the accuracy of a computed solution.

Many methods are known for estimating the condition of a linear system $Ax=b$ with $A\in\mathbb{R}^{n\times n}$ and $b\in\mathbb{R}^n$, and rigorous inequalities are known for many of them. Statistical condition estimation (SCE) has been introduced in [1] and applied to linear systems in [2]. By not focusing on rigorous inequalities, SCE has been found to be entirely as reliable as competing methods and, moreover,

[*]Correspondence to: J. Xia, Department of Mathematics, University of California, Los Angeles, CA 90095, U.S.A.
[†]E-mail: jxia@math.ucla.edu

is much better at respecting structures that may be found in many of the problems. Structured perturbations for linear systems have attracted much attention recently (see [3–5], for example). SCE provides a systematic way of estimating structured condition numbers. Moreover, SCE is every bit as competitive on efficiency grounds when specialized to linear systems. Application of the SCE method itself does not require linearity. Both componentwise and normwise condition estimates can be provided by SCE.

The idea of SCE can be illustrated with a general real-valued function $f:\mathbb{R}^r\to\mathbb{R}$. By Taylor's theorem, we have

$$f(p+\delta z)-f(p)=\delta d^{\mathrm{T}}z+O(\delta^2)$$

where δ is small, $\|z\|_2^2=1$, and $d^{\mathrm{T}}\equiv\nabla f(p)$ is the gradient of f. It is clear that the following inequality is true up to first order in δ

$$|f(p+\delta z)-f(p)|\approx\delta\|d\|_2 \tag{1}$$

That is, $\|d\|_2$ can be considered as a local condition number that gives a good approximation error $\delta\|d\|_2$ in numerical computations of $f(p)$ (with δ on the order of the machine epsilon $\varepsilon_{\mathrm{mach}}$). According to [1], if z is selected uniformly and randomly from the unit r-sphere S_{r-1} (denoted as $z\in\mathrm{U}(S_{r-1})$) and then normalized, then the expected value $E(|d^{\mathrm{T}}z|/w_r)$ is $\|d\|_2$, where w_r is the Wallis factor [1]. We can then use $v=|d^{\mathrm{T}}z|/w_r$ as a condition estimator. The accuracy is given by the probability relationship [1]:

$$\Pr(\|d\|_2/\gamma\leqslant v\leqslant\gamma\|d\|_2)\approx1-\frac{2}{\gamma\pi},\quad \gamma>1$$

To increase the accuracy, multiple samples of $z^{(j)}$ can be used [1]. For example, a 2-sample estimator is given by $v^{(2)}=(w_2/w_r)\sqrt{|d^{\mathrm{T}}z^{(1)}|^2+|d^{\mathrm{T}}z^{(2)}|^2}$, where $[z^{(1)},z^{(2)}]$ is orthonormalized after $z^{(1)}$ and $z^{(2)}$ are generated. The accuracy of $v_i^{(2)}$ is

$$\Pr(\|d\|_2/\gamma\leqslant v^{(2)}\leqslant\gamma\|d\|_2)\approx1-\frac{\pi}{4\gamma^2},\quad \gamma>1$$

The results can be extended to vector-valued functions $f:\mathbb{R}^r\to\mathbb{R}^n$. By Taylor's theorem, we have

$$\frac{f(p+\delta z)-f(p)}{\delta}=Dz+O(\delta)$$

where $D\equiv Df(p)=(\partial f_i/\partial p_j)\in\mathbb{R}^{n\times r}$ is the Fréchet derivative of f at p. (Later we will use $Df(p;z)=Df(p)z$ to denote the Fréchet derivative of f with respect to p evaluated in the direction of z.) Let $D\equiv[d_1,\ldots,d_n]^{\mathrm{T}}$. The SCE estimate for the ith entry of f is given by $v_i=|d_i^{\mathrm{T}}z|/w_r$. A 2-sample SCE estimate takes the form $v_i^{(2)}=(w_2/w_r)\sqrt{|d_iz^{(1)}|^2+|d_iz^{(2)}|^2}$. For convenience, for the entire solution vector, we can define the following 2-sample componentwise condition estimator:

$$\kappa_{\mathrm{csce}}^{(2)}=\frac{w_2}{w_r}\sqrt{|Dz^{(1)}|^2+|Dz^{(2)}|^2}$$

which is a condition vector for the n solution components. A general k-sample SCE estimator can be similarly defined.

In particular, we are interested in the situation where f is defined by the problem of solving a structured linear system. Assume that A and b are structured and depend on r parameters $p_1, \ldots, p_r$. We may assume a structure map

$$\tau(p) \equiv [\tau_A(p), \tau_b(p)] = [A, b], \quad p \in \mathscr{P} = \mathbb{R}^r \tag{2}$$

Note that τ can be either linear or nonlinear compared with linear or affine maps in [2]. (We say that the system is linearly or nonlinearly structured.) We exploit structured perturbations by perturbing each p_i to $p_i + \delta z_i$. We represent the problem of solving the linear system as

$$x = A^{-1}b = \tau_A(p)^{-1}\tau_b(p) \equiv f(p)$$

The condition of f at p can be determined by the Fréchet derivative $Df(p; z)$, which, according to Section 2.1, is given by $A^{-1}(D\tau_b(p; z) - D\tau_A(p; z)x)$. We then have the following k-sample structured SCE algorithm based on the multiple-sample SCE algorithm in [2].

Algorithm 1 (*k-sample componentwise structured SCE for* $x = A^{-1}b$)

1. Generate $z^{(1)}, \ldots, z^{(k)} \in \mathrm{U}(S_{r-1})$. Orthonormalize $[z^{(1)}, \ldots, z^{(k)}]$ to obtain $[\hat{z}^{(1)}, \ldots, \hat{z}^{(k)}]$. Replace each $z^{(i)}$ by $\hat{z}^{(i)}$.
2. Calculate $u^{(i)} = A^{-1}(D\tau_b(p; z^{(i)}) - D\tau_A(p; z^{(i)})x)$, $i = 1, \ldots, k$, where $x = A^{-1}b$ (usually an approximate solution).
3. Calculate the k-sample componentwise SCE condition vector:

$$\kappa_{\text{csce}}^{(k)} = \frac{w_k}{w_r}\sqrt{|u^{(1)}|^2 + \cdots + |u^{(k)}|^2} \tag{3}$$

On the other hand, SCE can also estimate the regular Frobenius-norm condition number $\kappa_{\mathrm{F}}(A) = \|A\|_{\mathrm{F}}\|A^{-1}\|_{\mathrm{F}}$ (nonstructured). It is shown in [6] that for a matrix $B \in \mathbb{R}^n$ the expected value of $\|Bu\|_2$, where $v \in \mathbb{R}^r$ has entries in $N(0, 1)$, is equal to $w_n\|B\|_{\mathrm{F}}$. Thus, by letting $B = A^{-1}$ we have the following algorithm [2].

Algorithm 2 (*k-sample SCE for estimating* $\kappa_{\mathrm{F}}(A) = \|A\|_{\mathrm{F}}\|A^{-1}\|_{\mathrm{F}}$)

1. Generate $z^{(1)}, \ldots, z^{(k)} \in \mathrm{U}(S_{r-1})$. Orthonormalize $[z^{(1)}, \ldots, z^{(k)}]$ to obtain $[\hat{z}^{(1)}, \ldots, \hat{z}^{(k)}]$. Replace each $z^{(i)}$ by $\hat{z}^{(i)}$.
2. Solve

$$Au^{(i)} = z^{(i)}, \quad i = 1, \ldots, k \tag{4}$$

3. Calculate the k-sample Frobenius-norm SCE condition estimate

$$\kappa_{\text{fsce}}^{(k)} = \frac{w_k}{w_n}\|A\|_{\mathrm{F}}\sqrt{\|u^{(1)}\|_2^2 + \cdots + \|u^{(k)}\|_2^2} \tag{5}$$

This algorithm is especially useful for large systems where traditional condition estimators can be expensive. Usually, $k = 1$ or 2 is sufficient for both Algorithms 1 and 2.

The purpose of this paper is to provide additional examples of structured (possibly large) linear systems, illustrate the performance of SCE on such systems, and urge the use of SCE when appropriate. We have two major tasks: one is to consider linear systems depending both linearly and nonlinearly on parameters and the other is to improve the efficiency of SCE for large systems by efficiently approximating the solution $u^{(i)}$ in (4) in various situations. MATLAB codes for SCE have been developed and are available from the authors.

The cost of a 1-sample SCE is generally no more than the work of solving the system. In fact, in direct solutions of the problem, a 1-sample SCE needs only one linear solve that is usually much faster than the matrix factorization. When iterative methods are used in SCE, the number of iterations is usually far less than that required for the solution, since only modest accuracy is desired. SCE is also cheaper than some other methods such as the one-cycle power method estimate of the 2-norm condition where two linear solves are needed.

This paper is organized as follows. In Section 2, we work on some general linearly or nonlinearly structured matrices. Examples including tridiagonal matrices, bidiagonal matrices, Vandermonde matrices, and Cauchy matrices are presented. Some linear system examples and Sylvester equation examples are presented. Section 3 discusses SCE based on some direct and iterative algorithms for large linear systems such as the conjugate gradient (CG) method, the multigrid method, and methods based on semi-separable representations.

2. SCE FOR STRUCTURED SYSTEMS

In many situations, such as when A is structured, it is more meaningful to consider the condition in terms of perturbations that respect the structure. Structured condition analysis has been involved in a lot of recent work, e.g. [3–5]. As an example, Higham and Higham [3] define the structured componentwise backward error and apply it to symmetric matrices and Toeplitz matrices as examples. Here, we provide more applications where SCE clearly reflects the structured conditions. All these matrices are defined on certain parameters.

2.1. General structured linear systems

SCE for structured matrices was introduced in [2]. Those matrices can be viewed as linear structure maps of certain parameters. Here, we look at general linearly or nonlinearly structured systems. Consider linear systems of the general form:

$$L(A,X)=B \tag{6}$$

where L is linear in A and X. (This also includes systems such as $AX+XA=B$. More general affine problems of the form $L(A_1,A_2,\dots A_m,X)=B$ can be considered similarly.) For convenience, we also represent the problem of solving the linear system as

$$X=g([A,B])=g(\tau(p))\equiv f(p)$$

Given A and B we wish to determine the condition of g. The matrices A and B are structured and depend on r parameters $p_1,\dots,p_r$. We may assume a structure map:

$$\tau(p)\equiv[\tau_A(p),\tau_B(p)]=[A,B],\quad p\in\mathscr{P}=\mathbb{R}^r$$

We perturb each p_i to $p_i + \delta z_i$. Correspondingly, X is perturbed to $X + \delta\hat{X}$. Here $\hat{X}$ is the desired Fréchet derivative of the problem.

The Fréchet derivative can be derived in the following way. The linear system (6) is perturbed to $L(\tau_A(p+\delta z), X+\delta\hat{X}) = \tau_B(p+\delta z)$. Using the approximation $\tau(p+\delta z) \approx \tau(p) + \delta D\tau(p; z)$, we have

$$L(\tau_A(p)+\delta D\tau_A(p;z), X+\delta\hat{X}) \approx \tau_B(p)+\delta D\tau_B(p;z) \tag{7}$$

Expand (7) by the linearity of L, divide by δ, and let $\delta\to 0$. We obtain the following Fréchet relation:

$$L(\tau_A(p), \hat{X}) + L(D\tau_A(p;z), X) = D\tau_B(p;z) \tag{8}$$

which provides $\hat{X}$.

In particular, for the linear system $Ax(=L(A,x)) = b$, (8) leads to

$$\hat{X} = A^{-1}(D\tau_b(p;z) - D\tau(p;z)x)$$

Here, without loss of generality, we assume that b is not structured and is directly perturbed to $b+\delta\hat{b}$. That is, $\tau(p) \equiv \tau_A(p) = A$, and the Fréchet derivative of $f(p) = x = A^{-1}b$ evaluated in the direction z is given by

$$Df(p;z) = A^{-1}(\hat{b} - D\tau(p;z)x) \tag{9}$$

Given a structured matrix, we can choose a τ and thus determine $D\tau(p;z)$ to obtain $Df(p;z)$. After we obtain the Fréchet derivative (9), SCE applies similarly as discussed in [2] (see Algorithm 1). In SCE, we preserve the matrix structure by working with perturbations in $\mathscr{P}$ and then mapping the Fréchet derivative $D\tau(p;z)$ to the matrix structure.

2.2. *SCE for large linearly structured matrices*

We first investigate some classical examples of large linearly structured matrices that have been used in other testing of condition estimators. Note that for linear structures $D\tau(p;z) = \tau(z)$ in (9). The SCE estimate is used to compute an approximate forward error, which is compared with the forward error `FERR` obtained by LAPACK [7]. We also discuss various other types of condition for the problems.

2.2.1. Upper triangular matrices. An important class of structured matrices is upper triangular matrices. They are basic to, e.g. LU factorizations.

Example 1
Let $A \in \mathbb{R}^{n\times n}$ be an upper triangular matrix generated by computing the QR factorization of a random matrix whose elements are uniformly distributed on $[-1, 1]$. We consider the condition of $x = A^{-1}b$, where b is generated by multiplying A by the exact solution $x = [1, \ldots, 1]^{\mathrm{T}}$.

Table I shows the actual forward error $\max_n |x-\hat{x}|$, the approximate errors (or error bounds) `FERR` (from LAPACK), and $\varepsilon_{\text{mach}} \max_n \kappa_{\text{csce}}^{(1)}$ (from SCE with δ in (1) of order $\varepsilon_{\text{mach}}$ which approximates the actual perturbation in numerical computations). Algorithm 1 is used to compute the SCE condition vector $\kappa_{\text{csce}}^{(k)}$ in (3). Since $\kappa_{\text{csce}}^{(k)}$ is a vector of n componentwise estimates, only

Table I. Errors for Example 1.

n	$\max_n \lvert x-\hat{x}\rvert$	`FERR`	$\varepsilon_{\text{mach}} \max_n \kappa_{\text{csce}}^{(1)}$
1000	8.4E−15	6.8E−11	5.3E−15
2000	1.6E−14	3.0E−10	1.5E−14
4000	1.9E−14	1.3E−9	3.0E−14

$\max_n \kappa_{\text{csce}}^{(k)}$ is given. We can see that `FERR` is usually too pessimistic. In fact, a regular triangular solver applied to such systems gives very accurate results. By contrast, SCE respects this structure by perturbing only the upper triangular part. SCE gives more realistic condition estimates. For example, for $n=4000$ we have $\max_n \kappa_{\text{csce}}^{(2)} = 1.4\times 10^2$. On the other hand, the Skeel condition number is $\kappa_{\text{skeel}} = 2.9\times 10^3$. In fact, the problem is considered to be ill-conditioned by condition estimators such as MATLAB routines `rcond` (which uses LAPACK condition estimators `xGECON` based on Higham's modification of Hager's method [8]) and `condest` (which is based on Hager's method and a block generalization by Higham and Tisseur [9]). Both 1/`rcond` and `condest` measure normwise conditions and can be conservative. For this example with $n=4000$, they both give an estimate of 5.4×10^7. SCE perturbs only the upper triangular part instead of the entire matrix as many other estimators do. Of course, this phenomenon is easily detected by a person. However, the difficulty arises when such a detection is 'automatic' as, for example, may happen if other software is doing the checking.

In addition, SCE is also very efficient. For $n=4000$, the SCE MATLAB routine takes only about $\frac{1}{5}$th the time of the MATLAB built-in functions `rcond` or `condest`.

2.2.2. Upper bidiagonal matrices. Even more dramatic is the case of, for example, upper bidiagonal matrices.

Example 2
Consider the $n\times n$ matrix A

$$A = \begin{bmatrix} 1 & 1 & & & 0 \\ & 1 & 1 & & \\ & & 1 & \ddots & \\ & & & \ddots & 1 \\ 0 & & & & 1 \end{bmatrix}, \quad A^{-1} = \begin{bmatrix} 1 & -1 & 1 & \cdots & (-1)^{n+1} \\ & 1 & -1 & & \vdots \\ & & 1 & \ddots & \vdots \\ & & & \ddots & -1 \\ 0 & & & & 1 \end{bmatrix}$$

As before, we look at the conditions of $x = A^{-1}b$, where b is generated by multiplying A with x whose entries are uniform random numbers on $[-1, 1]$.

The matrix A is used in [10] as a counterexample to show that Hager's method gives poor condition estimates. It is easy to see that $\kappa_1(A) = 2n$. Therefore, in this sense A is ill-conditioned for large n. This is further supported by the condition estimates from 1/ `rcond` and `condest`.

However, in practice $Ax=b$ can be accurately solved with backward substitution. This is precisely reflected by the SCE estimates.

SCE can maintain the structural restrictions in different ways. The matrix A can be defined in terms of different types of parameters. For example, we can consider A to be a general upper triangular matrix, a bidiagonal matrix, a general upper bidiagonal matrix, a Toeplitz matrix with only two nonzero diagonals, or a matrix defined by a single parameter (i.e. 1). Correspondingly, we have different choices of perturbations ΔA in SCE, where the entries of ΔA have absolute values bounded by the machine precision $\varepsilon_{\text{mach}}$:

(i) ΔA is an upper triangular matrix;
(ii) ΔA is an upper bidiagonal matrix;
(iii) ΔA has the following structure:

$$\Delta A = \begin{bmatrix} \Delta a_1 & \Delta a_2 & & 0 \\ & \Delta a_1 & \ddots & \\ & & \ddots & \Delta a_2 \\ 0 & & & \Delta a_1 \end{bmatrix}$$

(iv) ΔA has the following structure:

$$\Delta A = \begin{bmatrix} \Delta a & \Delta a & & 0 \\ & \Delta a & \ddots & \\ & & \ddots & \Delta a \\ 0 & & & \Delta a \end{bmatrix}$$

We compute SCE componentwise condition estimates $\kappa_{\text{csce}}^{(1)}$ with these choices of ΔA for different matrix sizes n. Table II presents the approximate forward errors.

We see that, with all choices of ΔA, the error estimate $\varepsilon_{\text{mach}} \max_n \kappa_{\text{csce}}^{(1)}$ is more reliable than FERR. Again, we can compare the condition estimates from different estimators. For $n=4000$, we

Table II. Errors for Example 2.

			$\varepsilon_{\text{mach}} \max_n \kappa_{\text{csce}}^{(1)}$			
n	$\max_n \lvert x-\hat{x}\rvert$	FERR	Choice (i)	Choice (ii)	Choice (iii)	Choice (iv)
1000	7.8E−16	8.1E−11	7.7E−14	1.4E−14	1.1E−14	4.9E−15
2000	8.9E−16	3.3E−10	1.2E−13	2.8E−14	3.4E−14	1.1E−14
4000	2.8E−15	2.9E−9	3.0E−13	2.2E−14	2.2E−14	2.1E−14
8000	3.4E−15	5.3E−9	1.3E−12	2.9E−14	2.7E−14	3.5E−14

have $\max_n \kappa_{\text{csce}}^{(1)}$=98.5 with choice (ii). On the other hand, the Skeel condition number is 6.6×10^3. Also `1/rcond` and `condest` return the estimates of 4.4×10^3 and 8.0×10^3, respectively.

This example illustrates that carefully chosen parameters are helpful for structured SCE. For the first choice of ΔA, the entire upper triangular part with $r=\frac{1}{2}n(n+1)$ entries is perturbed and the estimate is more conservative than those from the other three choices. That is, choices (ii), (iii), and (iv) respect the structures better. In addition, good choices of parameters can also save on cost and storage. SCE with choices (ii), (iii), or (iv) can be applied to large matrices for which other routines such as `1/rcond` and `condest` have trouble due to speed or memory issues. In general, choices (iii) and (iv) are more efficient than (ii).

2.2.3. Sylvester equation with triangular coefficient matrices. Consider the linear problem (6) where L is defined by two coefficient matrices $T\in\mathbb{R}^{m\times m}$ and $R\in\mathbb{R}^{n\times n}$:

$$L(T,R,X)=TX+XR$$

The system $L(T,R,X)=B$ is a Sylvester equation:

$$TX+XR=B \tag{10}$$

Sylvester equations play an important role in control theory, signal processing, invariant subspace computation, numerical solution of differential equations, and many others. Equation (10) can be rewritten as

$$A\,\text{vec}(X)=\text{vec}(B)$$

where $A=I_n\otimes T+R^{\mathrm{T}}\otimes I_m$, and vec$(\cdot)$ denotes the column vector formed by stacking the columns of a matrix from left to right.

Condition estimators for L or $A=I_n\otimes T+R^{\mathrm{T}}\otimes I_m$ such as the one in [11] compute a condition number for L or A. The cost is usually of the same order as the solution cost for the equation. Here we show that, with nearly the same amount of work (for solving an extra Sylvester equation and computing some matrix products), SCE can obtain a condition estimate matrix that indicates the sensitivity of the solution entries. In addition, SCE gives less conservative condition estimates, especially for structured problems. We consider (10) in a structured form, that is, T is lower triangular and R is upper triangular. Such a form is important in the Bartels–Stewart algorithm [12] for solving (10).

Example 3
Consider the example in [13].

$$T=\text{diag}(1,2,\ldots,m)+N_m,\quad R=2^{-t}I_n-\text{diag}(1,2,\ldots,n)+N_n^{\mathrm{T}}$$

where t is a scalar, and

$$N_k=\begin{bmatrix}0 & & & & 0\\ 1 & 0 & & & \\ 1 & 1 & 0 & & \\ \vdots & \vdots & & \ddots & \\ 1 & 1 & \cdots & 1 & 0\end{bmatrix}_{k\times k}$$

For simplicity, let $m=n$. It is clear that

$$\|L\|\|L^{-1}\| \equiv \|A\|_2\|A^{-1}\|_2 \gtrsim n\cdot 2^t$$

We choose $n=20$ and $t=20$ and generate B by letting X have uniformly random entries in $[0,1]$. Then this estimate gives a lower bound of 2.1×10^7. The actual condition number $\|A\|_2\|A^{-1}\|_2$ is 4.5×10^9. These results indicate the poor condition of the problem. However, we find that most of the solution entries can be computed quite accurately by numerical algorithms such as the Bartels–Stewart algorithm. See Figure 1 for the errors in the solution entries and the corresponding entrywise SCE condition numbers. SCE clearly reflects the sensitivity of the solution entries.

Example 4
Consider another Sylvester equation where T and R are generated as in Example 1. Generate B by letting X have uniformly random entries in $[0,1]$. In Table III, we report the maximum of the errors of the solution entries $\max|\text{vec}(X)-\text{vec}(\hat{X})|$, the maximum SCE condition number of the entries $\max\kappa_{\text{csce}}^{(1)}$, and the traditional condition number $\kappa_2(A)=\|A\|_2\|A^{-1}\|_2$. We can see that the traditional condition number is too conservative, whereas the SCE condition is more reasonable. In addition, if we look at the individual SCE condition numbers for the entries, we can observe

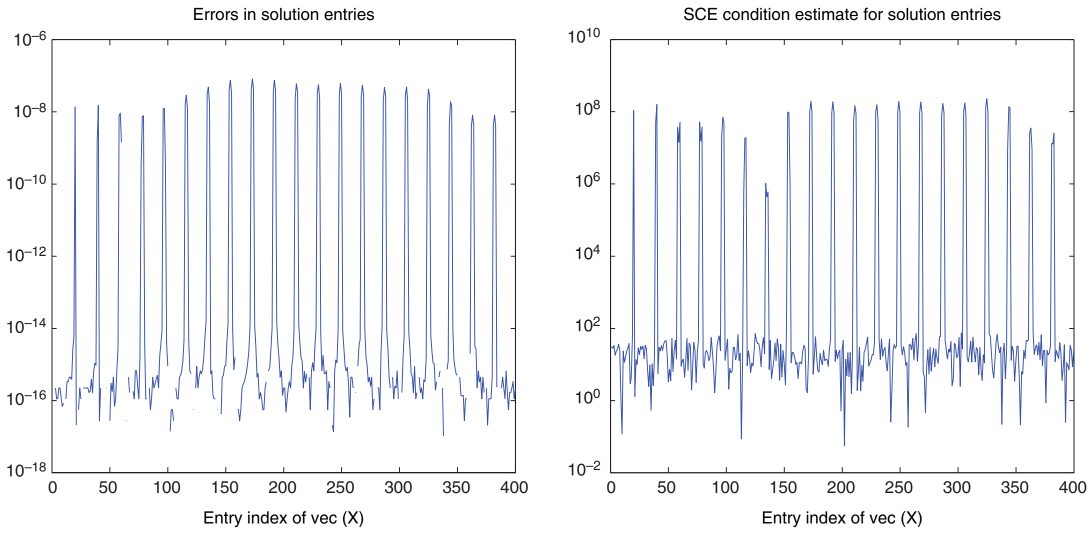

Figure 1. Errors in the entries of the numerical solution and the SCE condition estimate for the Sylvester system in Example 3.

Table III. Errors, SCE condition estimates, and traditional condition numbers for Example 4.

n	10	20	30	40	50
$\max\lvert\text{vec}(X)-\text{vec}(\hat{X})\rvert$	3.1E−12	7.2E−9	5.1E−8	5.9E−7	4.4E−6
$\max\kappa_{\text{csce}}^{(1)}$	5.2E3	1.7E6	3.2E7	9.4E10	2.4E11
$\kappa_2(A)$	6.1E6	5.4E10	6.1E11	7.4E12	1.3E15

a phenomenon similar to the previous example. That is, the SCE condition numbers reflect the accuracy of the entries much more accurately.

2.2.4. Other standard (structured) matrices. SCE can obtain similar results for many other common structured problems. The following are some examples:

1. An upper Hessenberg matrix H generated by computing the factorization $B = PHP^{\mathrm{T}}$ of a random matrix B whose elements are uniformly distributed on $[-1, 1]$, where P is an orthogonal matrix.
2. Pei's matrix [14]
$$A = \alpha I + vv^{\mathrm{T}}, \quad \alpha > 0, \quad v = [1, \ldots, 1]^{\mathrm{T}}$$
3. A symmetric Toeplitz matrix whose entries are uniformly distributed on $[-1, 1]$.

2.3. SCE for nonlinearly structured matrices

The structures discussed in [2] and also in Section 2.2 are linear maps of the parameters. In fact, SCE can also be applied to many linear systems with nonlinear structure maps. Here, we would like to investigate the componentwise sensitivity of solutions to some typical nonlinearly structured linear systems.

2.3.1. Vandermonde matrices. Consider the *Vandermonde* matrix

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ v_1 & v_2 & \cdots & v_n \\ v_1^2 & v_2^2 & \cdots & v_n^2 \\ \vdots & \vdots & & \vdots \\ v_1^{n-1} & v_2^{n-1} & \cdots & v_n^{n-1} \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{11}$$

defined by n distinct scalars $v_1, \ldots, v_n \in \mathbb{R}$, and the associated linear systems

$$\text{primary: } Vx = b$$

or

$$\text{dual: } V^{\mathrm{T}}x = b$$

The Vandermonde matrix is nonsingular according to the formula (see, e.g. [15])

$$\det V = \prod_{i,j,i>j} (v_i - v_j)$$

which is easy to verify. Vandermonde systems are traditionally considered to be very ill-conditioned. In fact, it is well known that the condition number $\kappa_\infty(V) = \|V\|_\infty \|V^{-1}\|_\infty$ has lower bounds growing exponentially in n as $O(2^n)$ [16]. However, accurate solutions to Vandermonde systems are still possible. Björck and Pereyra present some examples that can be solved with good accuracy by their algorithm [17]. Higham gives an error analysis of the Björck–Pereyra algorithm and shows why accurate solutions are possible [18].

When SCE is applied to the Vandermonde system $Vx=b$, we can evaluate the Fréchet derivative (9) easily. Note that $p_j \equiv v_j$ in (9) and $D\tau(p;z)\equiv D$ is now a matrix with entries

$$d_{ij}=\begin{cases}0 & \text{if } i=1\\ (i-1)v_j^{i-2}\hat{v}_j & \text{otherwise}\end{cases}$$

where each v_j is perturbed to $v_j+\delta\hat{v}_j$.

Example 5
Solve the problem $Vx=b$, where $v_j=j/n$, $j=1,\ldots,n$, and b is generated by $V^{\mathrm{T}}x$ with the entries of x being random numbers chosen from a normal distribution.

Although the matrix is very ill-conditioned for large n, the numerical solution $\tilde{x}$ obtained by Gaussian elimination with partial pivoting has good accuracy in certain entries. For example, for $n=15$, the matrix has $\kappa_\infty(V)$ on the order of 10^{12}. However, some entries of $\tilde{x}$ have up to 11 digits of accuracy. See Figure 2 for the errors in the solution entries and the corresponding componentwise SCE condition estimate. The componentwise SCE condition numbers precisely reflect the accuracy in the solution entries. Standard condition estimators such as the Skeel componentwise condition can be very conservative. Note that the Björck–Pereyra algorithm can be used in the estimations for solving Vandermonde systems with $O(n^2)$ complexity.

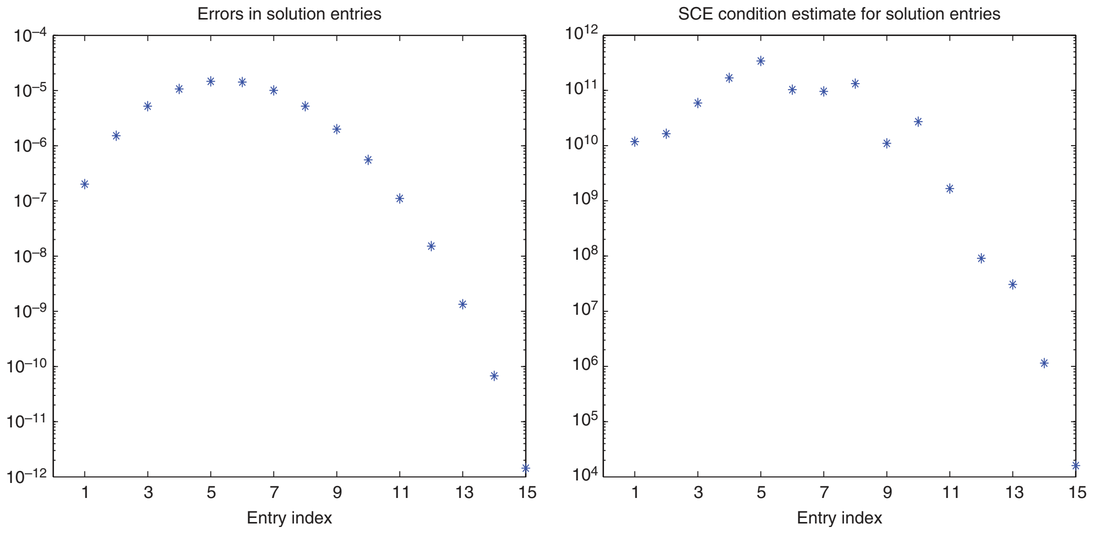

Figure 2. Errors in the entries of the numerical solution and the SCE condition estimate for a Vandermonde system with $n=15$ in Example 5.

2.3.2. Cauchy matrices. As another example now consider the *Cauchy* matrix

$$C = \left(\frac{1}{s_i - t_j}\right)_{1\leqslant i,j\leqslant n} \tag{12}$$

defined by $2n$ pairwise distinct parameters $s_i, t_j \in \mathbb{R}$. It is easy to verify that C is nonsingular by an explicit formula for its determinant (see, e.g. [19]):

$$\det C = \frac{\prod_{i<j}(s_i - s_j)\prod_{i<j}(t_i - t_j)}{\prod_{i,j}(s_i - t_j)}$$

With different choices of s_i and t_j, the structure of the C matrix can be a symmetric Hankel matrix, Hilbert matrix, or others. For these nonlinear structures, we can have an SCE procedure similar to the one in the previous section. This time $D\tau(p;z) \equiv D$ in the Fréchet derivative (9) is a matrix with entries

$$d_{ij} = -\frac{\hat{s}_i - \hat{t}_j}{(s_i - t_j)^2}$$

where the parameters s_i and t_j are perturbed to $s_i + \delta\hat{s}_i$ and $t_j + \delta\hat{t}_j$, respectively. Similarly, results can be obtained for a *Cauchy-like* matrix

$$\tilde{C} = \left(\frac{u_i v_j}{s_i - t_j}\right)_{1\leqslant i,j\leqslant n}$$

Example 6
Solve $Cx = b$, where C is defined by $s_i = i-1,\ t_j = -j,\ i,j = 1,\ldots,n$, and b is generated by Cx with the entries of x to be random numbers chosen from a normal distribution.

For this example, we observe a phenomenon very similar to that in Example 5. That is, again, the SCE componentwise estimates precisely reflect the accuracy in the solution entries. Since the results are very similar to those in Figure 2, we skip the details.

3. SCE ALGORITHMS FOR LARGE MATRICES

In this section we consider reducing the cost of SCE by efficiently approximating the solutions of related linear systems in Algorithms 1 and 2. For simplicity we consider system (4), which is used to estimate the Frobenius-norm condition number κ_{F}.

3.1. SCE with iterative methods

It is usually expensive to estimate directly the condition of large discretized problems. The efficiency of SCE for large linear systems is outlined in [2]. For the purposes of condition estimation, the cost for approximating the solution to (4) is usually small compared with the cost for accurate system solving. In an iterative method for solving (4), the error vector satisfies an equation $e_{k+1} = Me_k$ where $M = -D^{-1}N$ is defined by a *splitting* $A = D + N$. Usually the spectral radius $\rho(M)$ determines the speed of convergence of the method. In SCE we need only solve (4) accurately enough to approximate the norm of the true solution [2]. If we start with $u_0 = 0$, then

$\|e_k\|_2 \approx \rho^k \|e_0\|_2 \approx \rho^k \|u\|_2$. Therefore, to approximate $\|u\|$ to within a relative accuracy τ, we need to iterate only k steps such that $\rho^k < \tau$. For the purposes of SCE, a relatively large τ, say, 0.9 is sufficient. Clearly, the reduction in the accuracy τ by a square root leads to the saving of half of the iterations.

A model problem from the 1D Poisson equation with Dirichlet boundary conditions is considered in [2]. The discretized matrix $A \in \mathbb{R}^{n\times n}$ is a tridiagonal matrix:

$$A = \begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix} \tag{13}$$

In [2] the solution to $Au = z$ was illustrated with SOR. However, for such large sparse matrices we can use other methods with better convergence such as the CG method and the multigrid method [20]. We discuss each below.

3.1.1. SCE with the CG method. The CG method is widely used in solving linear systems with symmetric positive definite (SPD) coefficient matrices A. It is attractive for large problems such as discretized systems. A well-known bound for the error e_k in the solution after k steps of iterations is [21]

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leqslant 2\left(\frac{\sqrt{\kappa_2(A)}-1}{\sqrt{\kappa_2(A)}+1}\right)^k \tag{14}$$

where $\|e_k\|_A = \sqrt{e_k^{\mathrm{T}} A e_k}$. That is, in practice we have the following bound for the number of CG iterations needed to reduce the error by a factor ε:

$$k^* = \left\lceil \frac{1}{2}\sqrt{\kappa_2(A)} \ln \frac{2}{\varepsilon} \right\rceil \tag{15}$$

More extensive investigations of the convergence behavior of CG are available (e.g. [22–24]). As an example, when the matrix has small isolated eigenvalues, an improved bound is [25]

$$k^* = \left\lceil \frac{1}{2}\sqrt{\frac{\lambda_n}{\lambda_p}} \left(\ln\frac{2}{\varepsilon} + \sum_{i=1}^{r-1} \ln \frac{\lambda_r}{\lambda_i} \right) \right\rceil + (r-1)\left\lceil \sqrt{\frac{\lambda_n}{\lambda_r}} + 1 \right\rceil \tag{16}$$

where $\lambda_1 \leqslant \cdots \leqslant \lambda_r \leqslant \cdots \leqslant \lambda_n$ are the eigenvalues of A, and r is an integer (number of isolated small eigenvalues).

These error analyses give us a sense about the role that ε plays in the iterations. A smaller ε often requires more iterations. On the other hand, when CG is used in SCE a relatively large ε is usually sufficient, which leads to fast convergence.

For the model problem (13) with $n = 10^4$, the SCE method estimates $\|A^{-1}\|_{\mathrm{F}}$ by solving $Au = z$ in Algorithm 2. For such a vector z, Figure 3 displays the errors and norms of the numerical solutions after some CG iterations with initial vector $u_0 = 0$. The CG method is relatively slow in

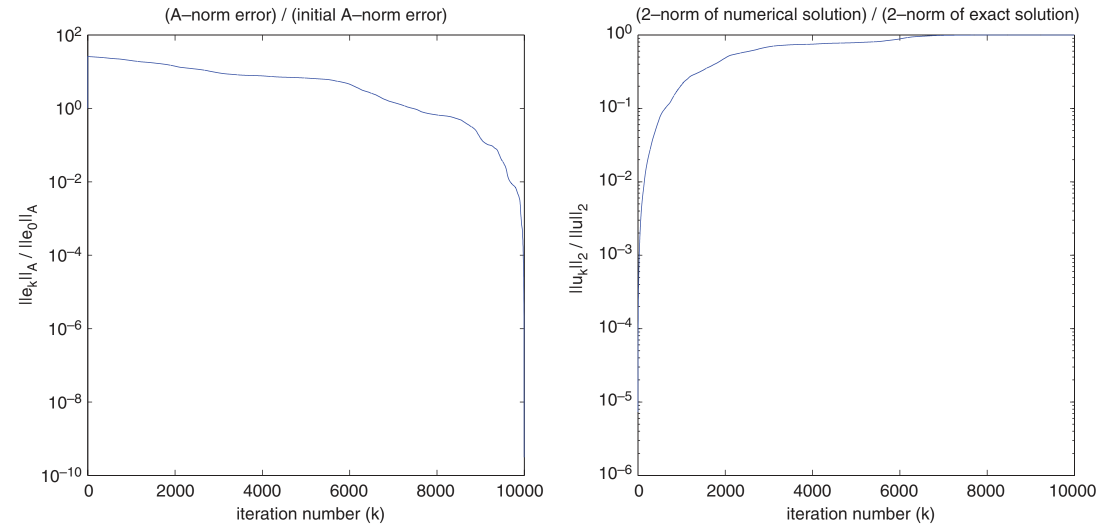

Figure 3. The CG method for the 1D discretized Poisson equation. The left figure shows the ratio $\|e_k\|_A/\|e_0\|_A$, and the right one shows the ratio $\|u_k\|_2/\|u\|_2$, in terms of the iterations.

reducing the error norm for this example. It takes $k=9147$ iterations to reach $\|e_k\|_A/\|e_0\|_A\leqslant 0.1$. On the other hand, it takes only $k=3065$ steps to estimate the norm $\|u_k\|_2$ to within a relative factor of about 70% of $\|u\|_2$. The corresponding estimate of $\kappa_{\mathrm{F}}(A)$ is $\kappa_{\mathrm{fsce}}^{(1)}=\|A\|_{\mathrm{F}}\|u_{3065}\|_2/\omega_n=1.48\times 10^9$ (Algorithm 2), which is a favorable approximation of the true condition number $\kappa_{\mathrm{F}}(A)=2.58\times 10^9$.

Example 7
Consider the matrix $A=\mathrm{diag}(\lambda_i)$ where

$$\lambda_1=10^{-4},\quad \lambda_2=10^{-2},\quad \lambda_3=1,\quad \lambda_j=1+(j-3)\frac{99}{n-3},\quad j=4,\ldots,n$$

This is an example with small isolated eigenvalues. It is used in [25] to compare the bound (16) with the actual number of iterations.

Again we choose a vector z as in Algorithm 2 and use CG to solve $Au=z$ for $n=9900$. According to bound (16) (with $r=3$) after approximately $k=108$ steps of CG iterations, the A-norm relative error of the solution can be reduced by $\varepsilon=0.1$. According to the numerical experiments, we observe that for $k=108$ the norm of the numerical solution is almost identical to the true norm, or, $\|u_k\|_2/\|u\|_2=1$. We obtain an estimate of $\kappa_{\mathrm{F}}(A)$ as $\kappa_{\mathrm{fsce}}^{(1)}=\|A\|_{\mathrm{F}}\|u_{108}\|_2/\omega_n=3.13\times 10^7$. The true condition number is $\kappa_{\mathrm{F}}(A)=5.77\times 10^7$.

In practice, for better convergence, the CG method is always used with preconditioning. To illustrate the performance of preconditioned CG (PCG) in SCE, we consider an example of the 2D discretized Poisson equation.

Example 8

Consider a model problem derived from the 2D Poisson equation with Dirichlet boundary conditions. The discretized matrix A is a block-tridiagonal matrix

$$A=\begin{bmatrix} T & -I & & 0 \\ -I & \ddots & \ddots & \\ & \ddots & \ddots & -I \\ 0 & & -I & T \end{bmatrix}, \quad T=\begin{bmatrix} 4 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 4 \end{bmatrix} \tag{17}$$

where $A\in\mathbb{R}^{n\times n}$, $T\in\mathbb{R}^{m\times m}$ with $n=m^2$.

Matrix A is positive definite with eigenvalues

$$\lambda_{i,j}=4\sin^2\left(\frac{i\pi}{2(n+1)}\right)+4\sin^2\left(\frac{j\pi}{2(n+1)}\right), \quad i,j=1,\ldots,m$$

and eigenvectors being the columns of a matrix $Z\otimes Z$ where

$$Z=\left(\sqrt{\frac{2}{n+1}}\sin\frac{ij\pi}{n+1}\right)_{1\leqslant i,j\leqslant m}$$

Numerical results for $n=10^4$ with PCG are presented in Figure 4, where the preconditioner is obtained by incomplete Cholesky factorizations with no fill-in.

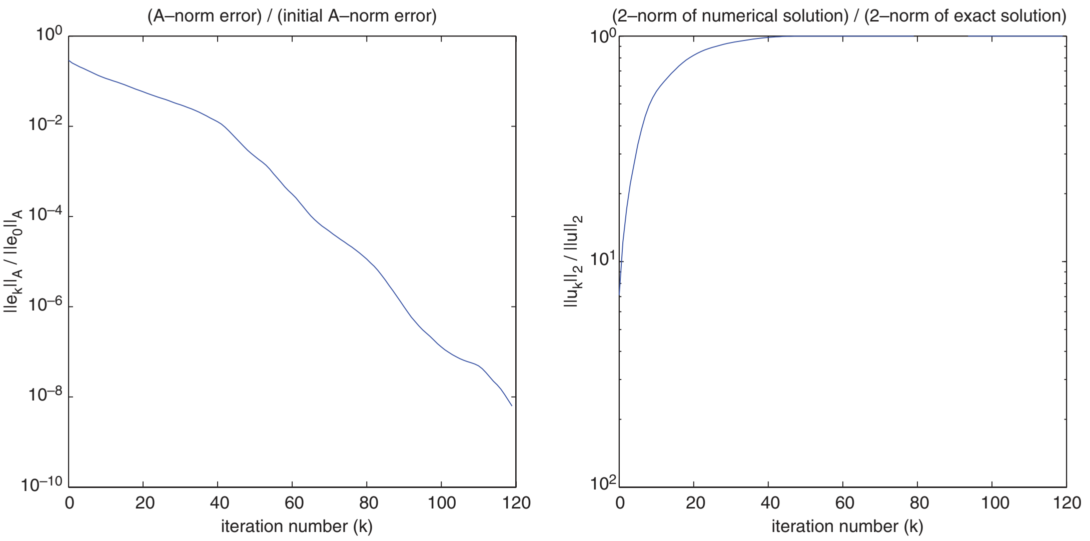

Figure 4. The CG method for the 2D discretized Poisson equation. The left figure shows the ratio $\|e_k\|_A/\|e_0\|_A$, and the right one shows the ratio $\|u_k\|_2/\|u\|_2$, in terms of the iterations.

The PCG method still takes $k=99$ steps to reach $\|e_k\|_A/\|e_0\|_A \leqslant 10^{-8}$, but for $k=17$ we already have $\|u_k\|_2/\|u\|_2 \geqslant 0.74$, and for $k=24$, $\|u_k\|_2/\|u\|_2 \geqslant 0.92$. The corresponding estimates of $\kappa_F(A)$ are 2.45×10^5 and 3.04×10^5, respectively, while the true value is $\kappa_F(A)=3.01\times 10^5$. To save on the cost in SCE, we can use a preconditioner computed by incomplete Cholesky factorizations with a large tolerance (say, 0.5).

Remark 1
Preconditioning can significantly improve the performance of CG in SCE as it can often accelerate the convergence of CG in the initial stages. It is observed that for ill-conditioned problems CG often converges slowly in its early stage (see, e.g. [23, 24]). A large $\kappa_2(A)$ may incur such a delay in convergence. In (16) the ε-independent term can be viewed as a bound for this delay. After preconditioning, the role of ε gets more significant, and the speedup in the early stage can be more noticeable.

Remark 2
The bounds on the number of CG iterations required in SCE are related to the condition of the problem, which on the other hand is what SCE tries to estimate. Therefore, during the iterations the numerical solutions can be used to obtain approximate bounds on the number of iterations required to reach a given accuracy. The following relationship between 2-norm condition numbers and Frobenius-norm condition numbers is then useful:

$$\frac{1}{n}\kappa_F(A) \leqslant \kappa_2(A) \leqslant \kappa_F(A)$$

This can be easily verified by noting

$$\frac{1}{\sqrt{n}}\|A\|_F \leqslant \|A\|_2 \leqslant \|A\|_F$$

3.1.2. SCE with the multigrid method. Multigrid was designed for PDEs and many other large problems and can obtain the solution in $O(n)$ operations for n unknowns. A multigrid V-cycle updates the errors such as

$$\begin{aligned} e_{k+1} &= R_w(I-P^{\mathrm{T}}(PAP^{\mathrm{T}})^{-1}PA)R_w e_k \\ &= Me_k \end{aligned}$$

where $R_w = 1 - wA/2$ with w a weight in the weighted Jacobi method, and P is a restriction matrix that maps from the current grid to a coarser grid. Multigrid is superior in that for a multigrid V-cycle with certain w, the spectral radius $\rho(M)<1$ is independent of grid sizes [26]. For example, when solving the Poisson equation with Dirichlet boundary conditions (model problem (17)) with $w=\frac{2}{3}$, we have $\rho(M)\equiv\frac{1}{9}$, which means, to approximate $\|u\|_2$ to within a relative accuracy of 89%, usually one multigrid V-cycle is sufficient. It is then clear that if multigrid is used with SCE, we do not need to compute the spectral radius $\rho(M)$ (as is done in [2]) to estimate the accuracy of the iterations.

In addition, noting that in a V-cycle the coarse grid sweeps are much faster than fine grid sweeps, we can reduce the number of weighted Jacobi iterations on fine grid sweeps (or all grid levels).

If we call the cost of weighted Jacobi iterations on the finest grid to be a *work unit* (WU), then the total cost of a V-cycle is [20]

$$2(1+2^{-2}+2^{-4}+\cdots+2^{-2n})\mathrm{WU}<\frac{2}{1-2^{-2}}\mathrm{WU} \tag{18}$$

This means that if we avoid certain weighted Jacobi iterations in the V-cycle we can save a large amount of the work.

The use of less weighted Jacobi iterations is acceptable because of the following. Assume that we totally avoid a Jacobi step R_w at the beginning or the end of M, that is, we have a new iteration matrix

$$\hat{M}=(I-P^{\mathrm{T}}(PAP^{\mathrm{T}})^{-1}PA)R_w \tag{19}$$

or

$$\hat{M}=R_w(I-P^{\mathrm{T}}(PAP^{\mathrm{T}})^{-1}PA) \tag{20}$$

It can be shown that $\rho(\hat{M})=\frac{1}{3}$ (see Appendix A.1). This indicates that one step of such a modified V-cycle is sufficient for condition estimation. The use of (19) or (20) instead of M saves approximately another half of the computational work (see (18)).

We use (19) with SCE for the matrix A in the above 1D model problem (13) and find that for $n=1023$ the SCE Frobenius-norm estimate for $\kappa_{\mathrm{F}}(A)$ is $\kappa_{\mathrm{fsce}}^{(2)}(A)=7.4\times10^6$, which approximates the true condition number $\kappa_{\mathrm{F}}(A)=8.7\times10^6$ very well. Note that in the regular multigrid method it takes about 10 full multigrid cycles to reach a relative accuracy of about 10^{-8}.

We further consider the 2D problem (17). One step of the modified V-cycle (19) with zero initial vectors is used in the SCE method when each linear system is solved. The method is implemented in MATLAB and compared with the routines `rcond` and `condest`. The timings for the routines on a Pentium IV 2.8 GHz desktop with 2GB memory are also presented (see Table IV).

Remark 3

In these results SCE slightly underestimates the true κ_{F} due to the choice of zero initial vectors in the multigrid method. This situation does not necessarily hold in general. We can show that under-estimations may occur, but they are not significant. For a detailed discussion, see Appendix A.2.

3.2. SCE with direct methods

In many situations the structures of matrices can be better exploited by direct methods. Sometimes direct solvers are used because it is hard to find effective preconditioners for iterative methods.

Table IV. Condition estimates and computation time (in seconds) for Example 8.

n	κ_1	1/`rcond`	1/`rcond` time	`condest`	`condest` time	$\kappa_{\mathrm{F}}(A)$	$\kappa_{\mathrm{fsce}}^{(2)}(A)$	SCE time
1089	6.8E2	6.8E2	3.1E−1	6.8E2	4.2E−2	1.1E4	4.4E3	2.4E−3
4225	2.6E3	2.6E3	1.5E1	2.6E3	3.9E−1	8.4E4	3.4E4	5.1E−3
16 641	Out of memory			1.0E4	3.6	6.4E5	2.7E5	2.4E−2
66 049	Out of memory			Out of memory		5.0E6	2.2E6	1.1E−1

In particular, direct solvers can be efficient in multiple-sample SCE where the same coefficient matrix is used to solve systems with different right-hand sides. Here, we take a look at a special type of direct method based on certain rank structures, and their applications to SCE.

3.2.1. SSS/HSS matrices. Chandrasekaran *et al.* propose two rank structures called *sequentially semi-separable* (SSS) structure and *hierarchically semi-separable* (HSS) structure [27, 28]. They consider matrices with small off-diagonal numerical ranks. They have shown the efficiency of these structures in solving many problems, especially large ones [27, 29].

Recently, they have developed strategies to maintain positive definiteness for SSS/HSS algorithms. These strategies are called *Schur-monotonic* approximations and they have been shown to give efficient preconditioners [30, 31]. However, they do not provide systematic estimations of the condition numbers in the algorithms. In fact, with SCE we can obtain normwise condition estimates very cheaply using the ideas of Schur-monotonic approximations of SPD matrices in [30, 31]. In the meantime, the method is guaranteed to run even for very large and ill-conditioned problems because the semi-separable approximations have the same or better positive definiteness compared with the original matrices.

Here we consider only the situation of HSS matrices. A 4×4 block HSS matrix looks like

$$A=\begin{bmatrix} D_1 & U_1B_1V_2^{\mathrm{T}} & U_1R_1B_3W_4^{\mathrm{T}}V_4^{\mathrm{T}} & U_1R_1B_3W_5^{\mathrm{T}}V_5^{\mathrm{T}} \\ U_2B_2V_1^{\mathrm{T}} & D_2 & U_2R_2B_3W_4^{\mathrm{T}}V_4^{\mathrm{T}} & U_2R_2B_3W_5^{\mathrm{T}}V_5^{\mathrm{T}} \\ U_4R_4B_6W_1^{\mathrm{T}}V_1^{\mathrm{T}} & U_4R_4B_6W_2^{\mathrm{T}}V_2^{\mathrm{T}} & D_4 & U_4B_4V_5^{\mathrm{T}} \\ U_5R_5B_6W_1^{\mathrm{T}}V_1^{\mathrm{T}} & U_5R_5B_6W_2^{\mathrm{T}}V_2^{\mathrm{T}} & U_5B_5V_4^{\mathrm{T}} & D_5 \end{bmatrix} \tag{21}$$

where the indices are defined following the rules in [29]. Matrix (21) is a simplified version of the original one proposed in [28]. The matrices D_i, U_i, R_i, B_i are called *generators*. For a general SPD matrix A, we can approximate it with a product $A=LL^{\mathrm{T}}$ where L is an HSS matrix, say, in 4×4 block form

$$L^{\mathrm{T}}=\begin{bmatrix} \tilde{D}_1 & \tilde{U}_1\tilde{B}_1\tilde{U}_2^{\mathrm{T}} & \tilde{U}_1\tilde{R}_1\tilde{B}_3\tilde{R}_4^{\mathrm{T}}\tilde{U}_4^{\mathrm{T}} & \tilde{U}_1\tilde{R}_1\tilde{B}_3\tilde{R}_5^{\mathrm{T}}\tilde{U}_5^{\mathrm{T}} \\ & \tilde{D}_2 & \tilde{U}_2\tilde{R}_2\tilde{B}_3\tilde{R}_4^{\mathrm{T}}\tilde{U}_4^{\mathrm{T}} & \tilde{U}_2\tilde{R}_2\tilde{B}_3\tilde{R}_5^{\mathrm{T}}\tilde{U}_5^{\mathrm{T}} \\ & & \tilde{D}_4 & \tilde{U}_4\tilde{B}_4\tilde{U}_5^{\mathrm{T}} \\ 0 & & & \tilde{D}_5 \end{bmatrix}$$

In general, the system $Ax=b$ can be solved (approximately) in $O(pn^2)$ flops, where n is the order of A, and p is a parameter related to A and the tolerance (p is usually the maximum of the off-diagonal numerical ranks). For the purposes of condition estimation, we can first obtain an approximation LL^{T} (very cheaply) and then find an approximate solution in $O(cn^2)$ complexity where $c\ll p$ (typically it is sufficient for c to be 2 or 3). This is because usually we need only the matrices B_i and R_i to have dimension 2 or 3 in order to obtain a good approximate solution. In fact, when the original system is solved with an HSS algorithm we can always obtain an approximation LL^{T} by dropping appropriate columns in the original HSS representations. This makes the cost for the estimation $O(c^2n)$ which is the cost for solving a compact HSS system [29]. This cost is usually minor compared with that of the system solve which costs $O(pn^2)$ for the factorization and $O(p^2n)$ for the substitution solve.

We can also set a large tolerance τ in the generation of HSS representations, instead of setting a fixed number of columns c. Readers are referred to [28, 29] for the details of HSS generations with tolerances.

Example 9
We consider a matrix $\tilde{A}$ obtained in the following way (we use the notation $\tilde{A}$ to mean we are considering a matrix other than the discretization matrix). Look at second-order finite-element discretizations of the following problem defined on the unit square:

$$a(x,y)\frac{\partial^2 u}{\partial x^2}+2b(x,y)\frac{\partial^2 u}{\partial x \partial y}+c(x,y)\frac{\partial^2 u}{\partial y^2}=f(x,y)$$

with

$$\begin{bmatrix} a(x,y) & b(x,y) \\ b(x,y) & c(x,y) \end{bmatrix} = \varepsilon I + dd^{\mathrm{T}}$$

where $\varepsilon>0$ and d is a unit vector. We assume a mixture of Dirichlet and Neumann boundary conditions. The problem is solved with the nested-dissection ordering of the nodes. We consider the condition of matrix $\tilde{A}$ which is the last Schur complement of the discretization matrix (we choose this matrix as it can be approximated by an HSS form with a small p). $\tilde{A}$ is dense and SPD.

We approximate $\tilde{A}$ by LL^{T} where L has R_i and B_i generators of order c. Then we estimate the Frobenius-norm condition number by SCE. In Table V we use $\kappa_{\mathrm{fsce}}^{(k)}(\tilde{A},c)$ to denote the SCE condition number with k samples and approximation $\tilde{A}\approx LL^{\mathrm{T}}$.

We can see that the approximation LL^{T} gives a very good estimate for the condition number. For $c=3$ the estimates are already close to those with the original matrix A.

Example 10
Consider the 2D discrete Poisson matrices (17). Similarly, we look at the condition of matrix $\tilde{A}$ which is the last Schur complement in nested dissection (Table VI). (Note that we are not considering the condition of the original A.)

Table V. Condition estimates and computatin time (in seconds) for Example 9.

n	ε	κ_{F}	$\kappa_{\mathrm{fsce}}^{(2)}(\tilde{A})$	$\kappa_{\mathrm{fsce}}^{(2)}(\tilde{A},c)$ $(c=2)$	$\kappa_{\mathrm{fsce}}^{(2)}(\tilde{A},c)$ $(c=3)$
200	1E−8	2.6E4	2.5E4	2.3E4	2.5E4
400	1E−4	5.2E6	5.6E6	1.0E6	5.2E6

Table VI. Condition estimates for Example 10.

n	κ_{F}	$\kappa_{\mathrm{fsce}}^{(2)}(\tilde{A})$	$\kappa_{\mathrm{fsce}}^{(2)}(\tilde{A},c)$ $(c=2)$	$\kappa_{\mathrm{fsce}}^{(2)}(\tilde{A},c)$ $(c=3)$
511	8.4E3	8.5E3	5.2E3	7.4E3
1023	2.4E4	2.7E4	1.0E4	1.4E4

It is also possible to consider the componentwise SCE for $Ax=b$, where A is an SSS/HSS matrix, and the entries of the generators are perturbed.

Remark 4
The structures in HSS matrices are not taken into account when considering the condition estimates, due to the difficulty in forming the Fréchet derivative $D\tau(p;z)$ in (9).

3.2.2. Large discretized matrices. We can extend the ideas in Section 3.2.1 to estimate the condition of large discretized problems from certain PDEs. Given a discretized problem we can convert it to a series of small intermediate problems by employing some well-known techniques such as the multifrontal method [32, 33] and nested dissection ordering [34, 35]. These intermediate problems are usually dense. Condition estimators based on standard solvers are still expensive. In [29] some superfast direct solvers have been developed for problems where the intermediate matrices have small off-diagonal numerical ranks.

Here, in order to obtain fast SCE estimation, we can use solvers similar to those in [29]. In addition, we use the approximation method in Section 3.2.1. Unlike the solvers in [29], for the purposes of condition estimation we usually do not require those intermediate problems to have small off-diagonal numerical ranks.

The cost of SCE using nested-dissection-based direct solvers is $O(n)$, where n is the number of unknowns. In fact, there are a total of $l=O(\log(n))$ levels in nested dissection. To solve each system, we can put the intermediate problems into two solve stages, separated by a switching level. Before the switching level, the problems are small and standard factorizations can be used. Assume that there are m levels before the switching level. The cost is then $\sum_{k=l-m+1}^{l} 4^{k-1}(O(2^{l-k}))^3 = O(4^l 2^m)$. After the switching level, structured factorizations (for $l-m$ levels) get involved. The cost is $\sum_{1}^{l-m} 4^{k-1} O(c^2 2^{l-k}) = O(4^l c^2/2^m)$, where the same notation c as in Section 3.2.1 is used to denote the maximum order of the appropriate generators in matrix approximations. The total cost of such an approximate solver is then $O(4^l 2^m)+O(4^l c^2/2^m)$, which can be approximately minimized with m satisfying $2^m \approx c$. Then the total cost is $O(4^l c)=O(cn)$. In this case, the costs before and after the switching level are approximately the same.

On the other hand, if the matrix is factorized (in $O(cn)$ time) when the problem is originally solved, then only $O((\log c)n)$ extra work is needed by SCE to estimate the condition. This is mainly the cost for the substitution solve based on the structured factors obtained in the factorization stage [29]. This also indicates another advantage of direct solvers, that is, a condition estimate can be obtained with only a little extra work based on the existing factorization.

Again we consider A in Example 8 and use SCE to estimate $\kappa_{\mathrm{F}}(A)$. When solving related linear systems we use the approximate solver mentioned above (superfast multifrontal method [29] with HSS approximations). Here, we set a tolerance τ. The k-sample SCE Frobenius-norm estimate is then denoted by $\kappa_{\mathrm{fsce}}^{(k)}(A,\tau)$. We observe from Table VII that even for a relatively large τ we can still obtain condition estimates that compare favorably with the true values.

Table VII. Condition estimates for Example 8 by superfast multifrontal methods with approximations.

n	κ_{F}	$\kappa_{\mathrm{fsce}}^{(2)}(A)$	$\kappa_{\mathrm{fsce}}^{(2)}(A,\tau)$ ($\tau=1\mathrm{E}{-1}$)	$\kappa_{\mathrm{fsce}}^{(2)}(A,\tau)$ ($\tau=1\mathrm{E}{-2}$)	$\kappa_{\mathrm{fsce}}^{(2)}(A,\tau)$ ($\tau=1\mathrm{E}{-3}$)
255^2	4.9E6	4.0E6	3.2E6	3.8E6	6.8E6
511^2	4.0E7	3.8E7	2.7E7	3.0E7	3.8E7

APPENDIX A

A.1. *Proof of $\rho(\hat{M})=\frac{1}{3}$ in Section 3.1*

We sketch the proof of $\rho(\hat{M})=\frac{1}{3}$, where $\hat{M}$ is the iteration matrix given by (19). For simplicity we consider the 1D case only, where $A\in\mathbb{R}^{n\times n}$ with $n=2^k-1$. Also, let $m=2^{k-1}-1$. Note that $m=(n+1)/2-1$.

Look at $\hat{M}$ in detail:

$$\hat{M}=[I-P^{\mathrm{T}}(PAP^{\mathrm{T}})^{-1}PA]R_{2/3}$$

with

$$P=\begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & & & & 0 \\ & & \frac{1}{2} & 1 & \frac{1}{2} & & \\ & & & \ddots & \ddots & \ddots & \\ 0 & & & & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix}_{m\times n}, \quad A=Z_n\Lambda Z_n^{\mathrm{T}}$$

where Λ is a diagonal matrix with diagonal entries $2(1-\cos j\pi/(n+1))$, $j=1,\dots,n$; $R_{2/3}=Z_n\Lambda_R Z_n\equiv Z_n(I-\frac{1}{3}\Lambda)Z_n$; and $Z_n=(\sqrt{2/(n+1)}\sin(ij\pi/(n+1)))_{1\leqslant i,j\leqslant n}$ is the eigenvector matrix of A satisfying $Z_n=Z_n^{\mathrm{T}}=Z_n^{-1}$. Similarly, let

$$Z_m=\left(\sqrt{\frac{2}{m+1}}\sin\left(\frac{ij\pi}{m+1}\right)\right)_{1\leqslant i,j\leqslant m}$$

Then it is easy to verify that $P=Z_m\Lambda_P Z_n$, where $\Lambda_P\in\mathbb{R}^{m\times n}$ is a matrix satisfying [26]

$$\lambda_{P;i,j}=\begin{cases} \frac{1}{\sqrt{8}}d_i\equiv\frac{1}{\sqrt{8}}\left(\cos\frac{\pi i}{n+1}+1\right) & \text{if } k=j \\ \frac{1}{\sqrt{8}}c_i\equiv\frac{1}{\sqrt{8}}\left(\cos\frac{\pi i}{n+1}-1\right) & \text{if } k=2^i-j \\ 0 & \text{otherwise} \end{cases} \tag{A1}$$

We can then express

$$\begin{aligned} T &= Z_n\hat{M}Z_n \\ &=\{I-(Z_nP^{\mathrm{T}}Z_m)[(Z_mPZ_n)(Z_nAZ_n)(Z_nP^{\mathrm{T}}Z_m)]^{-1}(Z_mPZ_n)(Z_nAZ_n)\}(Z_nR_{2/3}Z_n) \\ &=\{I-\Lambda_R^{\mathrm{T}}(\Lambda_P\Lambda\Lambda_P^{\mathrm{T}})^{-1}\Lambda_P\Lambda\}\Lambda_R \end{aligned} \tag{A2}$$

The matrix Λ_M has zeros on its main diagonal and antidiagonal. Therefore, one of its eigenvalues is its entry $t_{m+1,m+1}$, and the rest are formed by the eigenvalues of

$$\begin{bmatrix} t_{i,i} & t_{i,n+1-i} \\ t_{n+1-i,i} & t_{n+1-i,n+1-i} \end{bmatrix}, \quad i=1,\dots,m \tag{A3}$$

Firstly,

$$t_{m+1,m+1} = \lambda_{R;m+1,m+1} = 1 - \frac{1}{3}\cdot 2\left(1 - \cos\frac{(m+1)\pi}{n+1}\right) = \frac{1}{3} \tag{A4}$$

That is, $\frac{1}{3}$ is an eigenvalue of T and thus $\hat{M}$.

Secondly, we can compute the entries in (A3) explicitly as follows. The matrix $\Lambda_P \Lambda \Lambda_P^{\mathrm{T}}$ is a diagonal matrix with diagonal entries

$$f_i \equiv -\tfrac{1}{4}(c_i d_i^2 + c_i^2 c_{n+1-i}), \quad i = 1, \ldots, m$$

Noting $c_{n+1-i} = -d_i$, we have

$$f_i \equiv \tfrac{1}{4}(c_i^2 d_i - c_i d_i^2), \quad i = 1, \ldots, m \tag{A5}$$

Further computations show that

$$\begin{bmatrix} t_{i,i} & t_{i,n+1-i} \\ t_{n+1-i,i} & t_{n+1-i,n+1-i} \end{bmatrix} = \begin{bmatrix} \left(1 + \frac{1}{4}\frac{c_i d_i^2}{f_i}\right)\left(1 + \frac{2}{3}c_i\right) & \frac{1}{4}\frac{c_i c_{n+1-i} d_i}{f_i}\left(1 + \frac{2}{3}c_{n+1-i}\right) \\ \frac{1}{4}\frac{c_i^2 d_i}{f_i}\left(1 + \frac{2}{3}c_i\right) & \left(1 + \frac{1}{4}\frac{c_i^2 c_{n+1-i}}{f_i}\right)\left(1 + \frac{2}{3}c_{n+1-i}\right) \end{bmatrix}$$

$$i = 1, \ldots, m$$

By (A5), and again using $c_{n+1-i} = -d_i$,

$$\begin{aligned} \begin{bmatrix} t_{i,i} & t_{i,n+1-i} \\ t_{n+1-i,i} & t_{n+1-i,n+1-i} \end{bmatrix} &= \begin{bmatrix} \frac{1}{4}\frac{c_i^2 d_i}{f_i}\left(1 + \frac{2}{3}c_i\right) & -\frac{1}{4}\frac{c_i d_i^2}{f_i}\left(1 + \frac{2}{3}c_{n+1-i}\right) \\ \frac{1}{4}\frac{c_i^2 d_i}{f_i}\left(1 + \frac{2}{3}c_i\right) & -\frac{1}{4}\frac{c_i d_i^2}{f_i}\left(1 + \frac{2}{3}c_{n+1-i}\right) \end{bmatrix} \\ &= \begin{bmatrix} \frac{c_i}{c_i - d_i}\left(1 + \frac{2}{3}c_i\right) & -\frac{d_i}{c_i - d_i}\left(1 - \frac{2}{3}d_i\right) \\ \frac{c_i}{c_i - d_i}\left(1 + \frac{2}{3}c_i\right) & -\frac{d_i}{c_i - d_i}\left(1 - \frac{2}{3}d_i\right) \end{bmatrix} \end{aligned} \tag{A6}$$

whose eigenvalues are

$$\begin{aligned} \lambda_{\hat{M};i,1} &= 0 \\ \lambda_{\hat{M};i,2} &= \frac{c_i}{c_i - d_i}\left(1 + \frac{2}{3}c_i\right) - \frac{d_i}{c_i - d_i}\left(1 - \frac{2}{3}d_i\right) = 1 + \frac{2}{3}\frac{c_i^2 + d_i^2}{c_i - d_i} \end{aligned}$$

Using (A1) we have

$$\lambda_{\hat{M};i,2}=1+\frac{2}{3}\frac{2+2\cos^2\dfrac{\pi i}{n+1}}{-2}=1-\frac{2}{3}\left(1+\cos^2\frac{\pi i}{n+1}\right)$$

Clearly,

$$-\tfrac{1}{3}<\lambda_{\hat{M};i,2}<\tfrac{1}{3}$$

Therefore, $\hat{M}$ has one eigenvalue $\frac{1}{3}$, and the rest with absolute values bounded by $\frac{1}{3}$; thus $\rho(\hat{M})=\frac{1}{3}$.

A.2. Proof for Remark 3

We show that SCE with the modified multigrid V-cycle iteration may slightly underestimate the condition numbers, as mentioned in Remark 3.

The iteration scheme can be represented in the form

$$x_{i+1}=\hat{M}x_i+r$$

where $\hat{M}$ is given by (19). One step of the modified V-cycle with $x_0=0$ yields

$$x_1=r$$

On the other hand, the exact solution x satisfies

$$x=\hat{M}x+r$$

or

$$x=(I-\hat{M})^{-1}r$$

We would like to show $\|x_1\|_2\leqslant\|x\|_2$, or

$$\|(I-\hat{M})x\|_2\leqslant\|x\|_2$$

By (A2) this is equivalent to

$$\|(I-T)x\|_2\leqslant\|x\|_2$$

Recall that T is a matrix with nonzeros only on the main diagonal and antidiagonal, and the nonzero entries are explicitly given by (A6). Thus,

$$\|(I-T)x\|_2=\left|\sum_{i=1}^{m}\{[(1-t_{i,i})^2+t_{n+1-i,i}^2]x_i^2+[t_{i,n+1-i}^2 + (1-t_{n+1-i,n+1-i})^2]x_{n+1-i}^2\}+(1-t_{m+1,m+1})^2x_{m+1}^2\right|^{1/2}$$

According to (A4), $\left(1-t_{m+1,m+1}\right)^2x_{m+1}^2=\frac{4}{9}x_{m+1}^2\leqslant x_{m+1}^2$. Then if we can show $[(1-t_{i,i})^2+t_{n+1-i,i}^2]x_i^2\leqslant x_i^2$, $i=1,\dots,m$, or

$$(1-t_{i,i})^2+t_{n+1-i,i}^2\leqslant 1,\quad i=1,\dots,m$$

we are done. In fact,

$$(1-t_{i,i})^2+t_{n+1-i,i}^2=\left(\left(1+\frac{c_i}{2}\right)^2+\left(\frac{c_i}{2}\right)^2\right)\left(1+\frac{2}{3}c_i\right)^2$$
$$=\frac{1}{2}\left(1+\cos^2\frac{\pi i}{n+1}\right)\left(\frac{1}{3}+\frac{2}{3}\cos\frac{\pi i}{n+1}\right)^2$$

and $0\leqslant c_i\leqslant 1$ for $i=1,\ldots,m$; we can easily verify that

$$(1-t_{i,i})^2+t_{n+1-i,i}^2\leqslant\tfrac{1}{2}(1+1)(\tfrac{1}{3}+\tfrac{2}{3})^2=1$$

The above procedure also shows that $\|x_1\|_2$ is not much smaller than $\|x\|_2$, which means that although SCE with the modified V-cycle may possibly underestimate the condition number, it does not underestimate it by too much.

ACKNOWLEDGEMENTS

The authors wish to thank the editor and the referee(s) for their valuable comments.

REFERENCES

1. Kenney CS, Laub AJ. Small-sample statistical condition estimates for general matrix functions. *SIAM Journal on Scientific Computing* 1994; **15**:36–61.
2. Kenney CS, Laub AJ, Reese MS. Statistical condition estimation for linear systems. *SIAM Journal on Scientific Computing* 1998; **19**:566–583.
3. Higham DJ, Higham NJ. Backward error and condition of structured linear systems. *SIAM Journal on Matrix Analysis and Applications* 1992; **13**:162–175.
4. Gohberg I, Koltracht I. Mixed, componentwise, and structured condition numbers. *SIAM Journal on Matrix Analysis and Applications* 1993; **14**:688–704.
5. Rump SM. Structured perturbations. Part II: componentwise distances. *SIAM Journal on Matrix Analysis and Applications* 2003; **25**:31–56.
6. Gudmundsson T, Kenney CS, Laub AJ. Small-sample statistical estimates for matrix norms. *SIAM Journal on Matrix Analysis and Applications* 1995; **16**:776–792.
7. Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J, Du Croz J, Greenbaum A, Hammarling S, McKenney A, Sorensen D. *LAPACK Users' Guide* (3rd edn). SIAM: Philadelphia, PA, U.S.A., 1999.
8. Hager WW. Condition estimates. *SIAM Journal on Scientific Computing* 1984; **5**:311–316.
9. Higham NJ, Tisseur F. A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra. *SIAM Journal on Matrix Analysis and Applications* 2000; **21**:1185–1201.
10. Higham NJ. Algorithm 674: FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Transactions on Mathematical Software* 1988; **14**:381–396.
11. Byers RA. LINPACK-style condition estimator for the equation $AX-XB^{\mathrm{T}}=C$. *IEEE Transactions on Automatic Control* 1984; **29**:926–928.
12. Bartels RH, Stewart GW. Solution of the matrix equation $AX+XB=C$. *Communications of the ACM* 1972; **15**:820–826.
13. Golub GH, Nash S, Van Loan CE. A Hessenberg–Schur method for the problem $AX+XB=C$. *IEEE Transactions on Automatic Control* 1979; **24**:909–913.
14. Pei ML. A test matrix for inversion procedures. *Communications of the ACM* 1962; **5**:508.
15. Sharpe D. *Rings and Factorization*. Cambridge University Press: Cambridge, England, 1987.
16. Gautschi W, Inglese G. Lower bounds for the condition number of Vandermonde matrices. *Numerische Mathematik* 1988; **52**:241–250.

17. Björck Å, Pereyra V. Solution of Vandermonde systems of equations. *Mathematics of Computation* 1970; **24**: 893–903.
18. Higham NJ. Error analysis of the Björck–Pereyra algorithms for solving Vandermonde systems. *Numerische Mathematik* 1987; **50**:613–632.
19. Davis PJ. *Interpolation and Approximation*. Dover: New York, 1975.
20. Briggs WL, Henson VE, McCormick SF. *A Multigrid Tutorial* (2nd edn). SIAM: Philadelphia, PA, U.S.A., 2000.
21. Concus J, Golub GH, O'Leary DP. A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations. In *Sparse Matrix Computations*, Bunch JR, Rose DJ (eds). Academic Press: New York, 1976.
22. Axelsson O, Lindskog G. On the rate of convergence of the preconditioned conjugate gradient algorithm. *Numerische Mathematik* 1986; **48**:499–523.
23. Beckermann B, Kuijlaars ABJ. Superlinear convergence of conjugate gradients. *SIAM Journal on Numerical Analysis* 2001; **39**:300–329.
24. van der Sluis A, van der Vorst HA. The rate of convergence of conjugate gradients. *Numerische Mathematik* 1986; **48**:543–560.
25. Notay Y. On the convergence rate of the conjugate gradients in presence of rounding errors. *Numerische Mathematik* 1993; **65**:301–317.
26. Demmel J. *Applied Numerical Linear Algebra*. SIAM: Philadelphia, PA, U.S.A., 1997.
27. Chandrasekaran S, Dewilde P, Gu M, Pals T, Sun X, van der Veen A-J, White D. Some fast algorithms for sequentially semiseparable representations. *SIAM Journal on Numerical Analysis* 2005; **27**:341–364.
28. Chandrasekaran S, Gu M, Pals T. A fast *ULV* decomposition solver for hierarchically semiseparable representations. *SIAM Journal on Matrix Analysis and Applications* 2006; **28**:603–622.
29. Xia J. Fast direct solvers for structured linear systems of equations. *Ph.D. Thesis*, UC Berkeley, 2006.
30. Chandrasekaran S, Gu M, Li XS, Vesselevski P. Schur-monotonic semi-separable approximations of symmetric positive definite matrices. *Research Notes*, 2006.
31. Chandrasekaran S, Gu M, Li XS, Xia J. Schur-monotonic HSS approximations of symmetric positive definite matrices. *Research Notes*, 2006.
32. Duff IS, Reid JK. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Transactions on Mathematical Software* 1983; **9**:302–325.
33. Liu JWH. The multifrontal method for sparse matrix solution: theory and practice. *SIAM Review* 1992; **34**:82–109.
34. George JA. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis* 1973; **10**:345–363.
35. George JA, Liu JWH. *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall: Englewood Cliffs, NJ, 1981.