The Singular Value Decomposition: Its Computation and Some Applications

VIRGINIA C. KLEMA AND ALAN J. LAUB, MEMBER, IEEE

Abstract—**We provide a tutorial introduction to certain numerical computations both in linear algebra and linear systems in the context of bounded arithmetic. The essential characteristics of bounded arithmetic are discussed in an introductory section followed by a review of the fundamental concepts of numerical stability and conditioning. The singular value decomposition (SVD) is then presented along with some related comments on the numerical determination of rank. A variety of applications of the SVD in linear algebra and linear systems is then outlined. A final section discusses some details of the implementation of the SVD on a digital computer. An Appendix is provided which contains a number of useful illustrative examples.**

I. INTRODUCTION

IN THIS PAPER we shall address some numerical issues arising in the study of models described by linear systems of the familiar form

$$\dot{x}(t)=Ax(t)+Bu(t). \tag{1}$$

Here x is an n-vector of states and u is an m-vector of controls or inputs. This is one of the simplest models of a family of considerably more elaborate models. The interested reader is referred to [31] for an extended treatment and to [32] for a recent survey of geometric methods in linear multivariable control.

The emphasis here will be on numerical considerations pertinent to problems in linear systems theory. Many important problems have now been solved, frequently very elegantly, in linear control and systems theory. While these solutions are, for the most part, well-understood theoretically, very little is understood about their implementation on a digital computer.

Mathematically posed problems that are to be solved, or whose solution is to be confirmed on a digital computer must have the computations performed in the presence of (usually) inexact representation of the model or the problem itself. Furthermore, the computational steps must be performed in bounded arithmetic—bounded in the sense of finite precision and finite range. The finite precision means that computation must be done in the presence of rounding or truncation error at each stage. Finite range means that intermediate and final results must lie within the range of the arithmetic of the particular computing machine that is being used. Unfortunately, the range of arithmetic on many computing machines is not symmetric about zero. It is ever so likely that one will compute a small number lying in the range of finite arithmetic, a number which we shall call uflim, but not be able to compute 1/uflim without getting outside the finite range, i.e., overflowing.

One example of this situation is the inner product in orthogonal transformations, which, in the presence of near degeneracy in rank, tends toward the underflow limit, uflim, about 10^{-79} on IBM 360/370 machines. The overflow limit on these machines is about 10^{78}. Thus, the required reciprocal, 10^{79}, is outside the finite range. Particular care must be taken in software to cope with such problems.

The finite precision nature of computer arithmetic limits the number of digits available to represent the results of addition, subtraction, multiplication, and division and therefore makes unlikely that the associative and distributive laws hold for the actual arithmetic operations performed on the computing machine.

Numbers are usually represented in the computing machine in floating-point form which may be generalized in the following way. Let a number x be represented as

$$x\cong d\cdot\beta^{e} \tag{2}$$

where $-1<d<1$, β is the base of the floating-point arithmetic, e is an integer exponent, and $d=(d_1, d_2,\cdots,d_t)$ where t indicates the number of characters available to represent the fractional part of x. For decimal floating-point arithmetic this notation corresponds to standard scientific notation. The exponent part $e=(e_q, e_{q-1},\cdots,e_1)$ where q is the number of characters allocated to represent the exponent part of the number and therefore determines the range of arithmetic of the computing machine. Typical machine-dependent constants t, β, and q are given in [10] and [1]. We do not pursue further details here.

A typical machine representation of a floating-point number is

$$se_qe_{q-1}\cdots e_1 \quad d_1d_2\cdots d_t \tag{3}$$

where s is the sign of the number. Usually $0\leqslant e$, and the sign of the exponent is implicit in the sense that 0 repre-

Manuscript received April 12, 1979; revised August 2, 1979. Paper recommended by B. Francis, Chairman of the Linear Systems Committee. This work was supported in part by the U.S. Army Research Office and the Office of Naval Research under Contract DAAG29-79-C-0031 and in part by the National Science Foundation under Grant MCS 78-17697.

The authors are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139.

0018-9286/80/0400-0164$00.75

sents the smallest exponent permitted while $e_q e_{q-1} \cdots e_1$ with each $e_i = \beta - 1$ represents the largest possible exponent.

For example, assume a binary machine where $\beta = 2$ and $q = 7$. The bits representing the exponent part of the number range from 0000000 to 1111111. Both positive and negative exponents must be accomodated. The following table illustrates a typical way in which this is done:

Explicit Exponent (Binary)	Explicit Exponent (Decimal)	Actual Exponent (Decimal)
0000000	0	−64
⋮	⋮	⋮
0111111	63	−1
1000000	64	0
1000001	65	+1
⋮	⋮	⋮
1111111	127	+63

Note that the above range of actual exponents is not symmetric about zero. We say that the explicit exponent is the actual exponent excess 64.

Turning now to the fractional part of a floating-point number it is important to realize that computing machines do not generally perform a proper round on representing numbers after floating-point operations.

For example, truncation of the six digit number 0.367879 gives 0.36787, whereas proper round gives 0.36788. Such truncation can result in a bias in the accumulation of rounding errors and is an essential component in rounding error analysis.

One cannot assume that decimal numbers are correctly rounded when they are represented in bases other than 10, say, 2 or 16. In the sense that a number a is represented in the computing machine in floating point as a^*, the associated relative error in its representation is $|(a^* - a)/a| = \epsilon$ where ϵ, in general, is the relative precision of the finite arithmetic, i.e., the smallest number for which the floating-point representation of $1 \pm \epsilon$ is not equal to 1. We shall use the notation $\mathrm{fl}(\cdot)$ to denote floating-point computation so that we have

$$\epsilon := \min_{\delta} \{\delta: \mathrm{fl}(1 \pm \delta) \neq 1\}. \tag{4}$$

(Occasionally, ϵ is defined to be the largest number for which $\mathrm{fl}(1 \pm \epsilon) = 1$.) The number ϵ varies, of course, depending on the computing machine and arithmetic precision (single, double, etc.) that is used.

Let the floating-point operations, add, subtract, multiply, and divide for the quantities x_1 and x_2 be represented by $\mathrm{fl}(x_1 \,\mathrm{op}\, x_2)$. Then, usually,

$$\mathrm{fl}(x_1 \text{ op } x_2) = x_1(1 + \epsilon_1) \text{ op } x_2(1 + \epsilon_2) \tag{5}$$

where ϵ_1 and ϵ_2 are of order ϵ. Therefore, one can say, in many cases, that the computed solution is the exact solution of a neighboring, or a perturbed problem.

This last statement is a form of what is called backward error analysis. Of course, it would be aesthetically pleasing to account for roundoff errors "in a forward sense." However, because of possible problems in representing the "exact answer" (e.g., 1/3 or 1/10 on a binary machine) or because of the complicated nature of arithmetic operations for many matrix calculations, some form of backward error analysis is required. The term backward error analysis was identified and explicitly described by Wilkinson in [29] and [30] although the idea was implicit in the work of Givens [13].

Explicit considerations of roundoff error, underflows, and overflows, as well as formal treatment of numerical stability and conditioning which we shall discuss in Section II, have heretofore been largely ignored by the control/systems community and research by that community into stable, efficient, and reliable algorithms and their embodiment in robust mathematical software is only now in its infancy. The time has arrived when "amateurs" can no longer duplicate, for many classes of problems, the efforts of numerical and software experts (for example, [25], [12], [6], [8]). Many problems in control and systems theory are of such a form as to benefit directly from those efforts now, while, conversely, many problems in control/systems suggest exciting new avenues of numerical and software research. While it is a slow process, we are now just beginning to see some of the material (well-known to numerical analysts) presented in this paper filter down to the undergraduate curriculum in mathematics and engineering. This process is certain to have a significant impact on the future directions and development of control and systems theory and applications.

Before proceeding we shall standardize some notation to be used in this paper.

$\mathbb{F}^{m\times n}$	the set of all $m \times n$ matrices with coefficients in the field $\mathbb{F}$ ($\mathbb{F}$ will generally be $\mathbb{R}$ or $\mathbb{C}$)
$\mathbb{F}_r^{m\times n}$	the set of all $m \times n$ matrices of rank r with coefficients in the field $\mathbb{F}$
A^T	the transpose of $A \in \mathbb{R}^{m\times n}$
A^H	the conjugate transpose of $A \in \mathbb{C}^{m\times n}$
$\|A\|$	the spectral norm of A (i.e., the matrix norm subordinate to the Euclidean vector norm: $\|A\| = \max_{\|x\|_2 = 1} \|Ax\|_2$)
$\|A\|_F$	the Frobenius norm of $A \in \mathbb{C}^{m\times n}$; $\|A\|_F = (\sum_{i=1}^{m} \sum_{j=1}^{n} \|a_{ij}\|^2)^{1/2}$
$\mathrm{diag}(a_1, \cdots, a_n)$	the diagonal matrix $\begin{bmatrix} a_1 & & 0 \\ & \ddots & \\ 0 & & a_n \end{bmatrix}$
$\sigma(A)$	the spectrum of A
$A \geqslant 0$ [>0]	the symmetric (or Hermitian) matrix A is nonnegative (positive) definite.

II. Numerical Stability and Conditioning

In this section we give a brief discussion of two concepts of fundamental importance in numerical analysis:

1) numerical stability of an algorithm and
2) the condition, or perturbation analysis, of the problem posed.

While this material is standard in introductory textbooks such as [7], [27], or [10] we present it here both for completeness and because the two concepts are occasionally confused in the control/system literature.

Suppose we have some mathematically defined problem represented by f which acts on data $d \in \mathcal{D}$ = some set of data, to produce a solution $f(d) \in \mathcal{S}$ = some set of solutions. These notations are kept deliberately vague for expository purposes. Given $d \in \mathcal{D}$ we desire to compute $f(d)$. Frequently, only an approximation d^* to d is known and the best we could hope for is to calculate $f(d^*)$. If $f(d^*)$ is "near" $f(d)$ the *problem* is said to be *well-conditioned*. If $f(d^*)$ may potentially differ greatly from $f(d)$ when d^* is near d, the problem is said to be ill-conditioned or ill-posed. Again the concept "near" cannot be made precise without further information about a particular problem. Determining condition is based on the mathematical perturbation theory for the problem at hand.

An *algorithm* to determine $f(d)$ is *numerically stable* if it does not introduce any more sensitivity to perturbation than is already inherent in the problem. Stability ensures that the computed solution is near the solution of a slightly perturbed problem. More precisely, let f^* denote an algorithm used to implement or approximate f. Then f^* is stable if for all $d \in \mathcal{D}$ there exists $d^* \in \mathcal{D}$ near d such that $f(d^*)$ (the solution of a slightly perturbed problem) is near $f^*(d)$ (the computed solution, assuming d is representable in the computing machine; if d is not exactly representable we need to introduce into the definition an additional d^{**} near d but the essence of the definition of stability remains the same).

Of course, one cannot expect a stable algorithm to solve an ill-conditioned problem any more accurately than the data warrant but an unstable algorithm can produce poor solutions even to well-conditioned problems. There are thus two separate factors to consider in determining the accuracy of a computed solution $f^*(d)$. First, if the algorithm is stable $f^*(d)$ is near $f(d^*)$ and second, if the problem is well-conditioned $f(d^*)$ is near $f(d)$. Thus $f^*(d)$ is near $f(d)$.

To illustrate the above concepts suppose $A \in \mathbb{R}_n^{n\times n}$, $b \in \mathbb{R}^{n\times 1}$ and we seek a solution of the linear system of equations

$$Ax = b. \tag{5}$$

The computed solution x is obtained from the perturbed problem

$$(A+E)\hat{x} = b+\delta \tag{6}$$

where E represents the perturbation in A and δ represents the perturbation in the right-hand side b. Stable algorithms, programmed as portable reliable software [8] are readily available. The rounding error analysis for the linear equations problem (i.e., bounds on $\|E\|$ for some appropriate norm $\|\cdot\|$) can be read in [29], [11], [27] and we do not pursue the details here. The problem is said to be ill-conditioned (with respect to a norm $\|\cdot\|$) if $\|A\|\cdot\|A^{-1}\|$ is "large." There then exist right-hand sides b such that for a nearby b^* the solution corresponding to b^* may be as far away as $\|A\|\cdot\|A^{-1}\|$ from the solution corresponding to b.

One final comment is in order for this example. It would be convenient, and it is sometimes true, if in the definition of stability we could say that the computed solution is the exact solution of a slightly perturbed problem. However, there are some pathological problems for which one can only say that the computed solution is near the solution of a neighboring problem. Such solutions can arise in the solution of linear systems of equations of the type considered above and this fact was pointed out by Kahan [20].

Further illustration of the concepts of stability and condition will be found in the next section.

III. Singular Value Decomposition and Numerical Rank

One of the basic and most important tools of modern numerical analysis, particularly numerical linear algebra, is the singular value decomposition. The SVD was established for real square matrices in the 1870's by Beltrami and Jordan (see, e.g., MacDuffee [23, p. 78]), for complex square matrices by Autonne [2], and for general rectangular matrices by Eckart and Young [9] (the Autonne–Eckart–Young theorem). For a survey of the theory for compact operators, see Gohberg and Krein [14]. We shall define the SVD here and make a few comments about its properties and computation. In Section IV we shall see how the SVD can be used to compute reliably a number of the basic geometric objects of linear algebra. This is but one of many fields of application and it is likely that within five or ten years SVD will be one of the most important and fundamental working tools for the control/systems community, particularly in the area of linear systems.

We now state the SVD theorem. Square brackets will denote the complex case.

Theorem: Let $A \in \mathbb{R}_r^{m\times n}$ [$\mathbb{C}_r^{m\times n}$]. Then there exist orthogonal [unitary] matrices $U \in \mathbb{R}^{m\times m}$ [$\mathbb{C}^{m\times m}$] and $V \in \mathbb{R}^{n\times n}$ [$\mathbb{C}^{n\times n}$] such that

$$A = U\Sigma V^T [\, U\Sigma V^H] \tag{7}$$

where

$$\Sigma = \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}$$

and $S = \operatorname{diag}(\sigma_1, \cdots, \sigma_r)$ with

$$\sigma_1 \geqslant \cdots \geqslant \sigma_r > 0.$$

Proof: Since $A^TA \geqslant 0$ we have $\sigma(A^TA) \subseteq [0, +\infty)$.

Denoting $\sigma(A^TA)$ by $\{\sigma_i^2, i=1,\cdots,n\}$ we can arrange that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 = \sigma_{r+1} = \cdots = \sigma_n$. Let $v_1,\cdots,v_n$ be a corresponding set of orthonormal eigenvectors and let

$$V_1 = (v_1,\cdots,v_r)$$
$$V_2 = (v_{r+1},\cdots,v_n).$$

Then if $S = \text{diag}(\sigma_1,\cdots,\sigma_r)$ we have $A^TAV_1 = V_1S^2$ whence

$$S^{-1}V_1^TA^TAV_1S^{-1} = I. \tag{8}$$

Also $A^TAV_2 = V_2 \cdot 0$ so that $V_2^TA^TAV_2 = 0$ and thus $AV_2 = 0$. Let $U_1 = AV_1S^{-1}$. Then from (8) we have $U_1^TU_1 = I$. Choose any U_2 such that $U = (U_1, U_2)$ is orthogonal. Then

$$U^TAV = \begin{pmatrix} U_1^TAV_1 & U_1^TAV_2 \\ U_2^TAV_1 & U_2^TAV_2 \end{pmatrix}$$
$$= \begin{pmatrix} S & 0 \\ U_2^TU_1S & 0 \end{pmatrix}$$
$$= \begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix} = \Sigma$$

and so $A = U\Sigma V^T$ as desired.

The numbers $\sigma_1,\cdots,\sigma_r$ together with $\sigma_{r+1}=0,\cdots,\sigma_n=0$ are called the *singular values* of A and they are the positive square roots of the eigenvalues (which are nonnegative) of $A^TA[A^HA]$. The columns of U are called the *left singular vectors* of A (the orthonormal eigenvectors of $AA^T[AA^H]$) while the columns of V are called the *right singular vectors* of A (the orthonormal eigenvectors of $A^TA[A^HA]$). The matrix $A^T[A^H]$ has m singular values, the positive square roots of the eigenvalues of $AA^T[AA^H]$. The r (=rank (A)) nonzero singular values of A and $A^T[A^H]$ are, of course, the same. The choice of $A^TA[A^HA]$ rather than $AA^T[AA^H]$ in the definition of singular values is arbitrary. Only the potentially nonzero singular values will be of any real interest.

It is not generally a good idea to compute the singular values of A by first finding the eigenvalues of A^TA, tempting as that is. Consider the following example [17] with μ a real number with $|\mu| < \sqrt{\epsilon}$ (so that $\text{fl}(1+\mu^2)=1$). Let

$$A = \begin{bmatrix} 1 & 1 \\ \mu & 0 \\ 0 & \mu \end{bmatrix}.$$

Then

$$\text{fl}(A^TA) = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

so we compute $\hat{\sigma}_1 = \sqrt{2}$, $\hat{\sigma}_2 = 0$ leading to the (erroneous) conclusion that the rank of A is 1. Of course, if we could compute in infinite precision we would find

$$A^TA = \begin{pmatrix} 1+\mu^2 & 1 \\ 1 & 1+\mu^2 \end{pmatrix}$$

with $\sigma_1 = \sqrt{2+\mu^2}$, $\sigma_2 = |\mu|$ and thus rank $(A)=2$. The point is that by working with A^TA we have unnecessarily introduced μ^2 into the computations.

Fortunately, Golub and Reinsch [17] have developed an extremely efficient and stable algorithm (based largely on [15]) for computing the SVD which does not suffer from the above defect. The computed U and V are orthogonal to approximately the working precision and the computed singular values can be shown to be the exact σ_i's for $A+E$ where $\|E\|/\|A\|$ is a modest multiple of ϵ. Furthermore, since the σ_i^2's are, in fact, the eigenvalues of a symmetric matrix, they are guaranteed to be well-conditioned so that, with respect to accuracy, we are in the best of possible situations.

A fairly sophisticated implementation of this algorithm can be found in [12] (SVD and MINFIT). There are other SVD subroutines around and many have severe problems even if coded directly from [17]. One is probably best off using the version implemented in LINPACK [8], or the version in [12]. Using the SVD is more than just calling a subroutine, however, and we shall indicate, in Section V, some of the details of computing the SVD.

It is clear from the definition that the number of nonzero singular values of A determines its rank. While the question is not nearly as clear-cut in the context of computation on a digital computer, it is now generally acknowledged that the singular value decomposition is the only generally reliable method of determining rank numerically (see [16] for a more elaborate discussion).

Only rather recently has the problem of numerical determination of rank been well-understood. One recent treatment of the subject, including a careful definition of numerical rank, is a paper by Golub *et al.* [16]. The essential idea is as follows. We are going to look at the "smallest nonzero singular value" of a matrix A. Since that computed value is exact for a matrix near A it makes sense to consider the rank of all matrices in some δ-ball (w.r.t. the spectral norm $\|\cdot\|$, say) around A. The choice of δ may also be based on measurement errors incurred in estimating the coefficients of A or the coefficients may be uncertain because of roundoff errors incurred in a previous computation to get them. See [16] for further details.

In any case it can easily be shown that all matrices B lying strictly inside the σ_r-ball around A have rank $\geq r$. The matrix $B = U\hat{\Sigma}V^T$ where

$$\hat{\Sigma} = \begin{pmatrix} \hat{S} & 0 \\ 0 & 0 \end{pmatrix}$$

with $\hat{S} = \text{diag}(\sigma_1,\cdots,\sigma_{r-1})$ is a matrix with rank $r-1$ and $\|B-A\| = \sigma_r$. Thus if we choose as some "zero threshold" a number $\delta < \sigma_r$, we will consider A to have numerical rank r. There can sometimes be real difficulties in determining a "gap" between the computed last nonzero singular value and what should effectively be considered "zero." Unless a problem originator gives information about the certainty of his data we normally define the gap with respect to uncertainty in the data as a function of the

precision limit in the machine. Note that the ratio σ_1/σ_r cannot exceed the precision limit. Further details are found in [16].

The key quantity in rank determination is obviously σ_r. Moreover, this number gives a dependable measure of how far (in the $\|\cdot\|$ sense) a matrix is from matrices of lesser rank. But σ_r alone is clearly sensitive to scale so that a better measure is $\sigma_r/\|A\|$. But $\|A\|=\sigma_1$ so the important quantity is σ_r/σ_1 which turns out to be the reciprocal of the number $\kappa(A)=\|A\|\cdot\|A^+\|$, the so-called "condition number of A w.r.t. pseudoinversion." In the case when A is invertible, $\kappa(A)=\|A\|\cdot\|A^{-1}\|$ is the usual spectral condition number w.r.t. inversion. The use of $\kappa(A)$ as a condition number in the general case is actually somewhat more complicated. For a discussion of this and related matters the reader is urged to consult Stewart [28].

In solving the linear system $Ax=b$, the condition number $\kappa(A)=\|A\|\cdot\|A^{-1}\|$ gives a measure of how much errors in A and/or b may be magnified in the computed solution. Moreover, if $A\in\mathbb{R}_n^{n\times n}$, $\sigma_n(A)$ gives a measure of the "nearness" of A to singularity. A standard introduction to the solution of linear systems is the classical book of Forsythe and Moler [11] to which we further refer the interested reader. Note, however, that in this special case of invertible A there is nothing special about the spectral norm and $\sigma_n(A)$. In fact $1/\|A^{-1}\|$ measures the nearness of A to singularity in any matrix norm $\|\cdot\|$ and for certain norms it is easy to construct explicitly a matrix E with $\|E\|=1/\|A^{-1}\|$ and $A+E$ singular. In the nonsquare case things are not quite so straightforward except in the case of the spectral norm.

That other methods of rank determination are potentially unreliable is demonstrated by the following special case of a family of matrices considered by Ostrowski [24]. The near-singularity properties of this special case (actually its negative) were emphasized by Kahan [21]. Consider the matrix $A\in\mathbb{R}^{n\times n}$ whose diagonal elements are all -1, whose upper triangle elements are all $+1$, and whose lower triangle elements are all 0. This matrix is clearly of rank n, i.e., is invertible. It has a good "solid" upper triangular shape. All of its eigenvalues (all $=-1$) are well away from zero. Its determinant is $(-1)^n$—definitely not close to zero. But this matrix is, in fact, very near singular and gets more nearly so as n increases. Notice, for example, that

$$\begin{bmatrix} -1 & & +1 & \cdots & +1 \\ & \ddots & & \ddots & \vdots \\ & & \ddots & & \vdots \\ & & & & +1 \\ & & & & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1/2 \\ \vdots \\ 1/2^{n-1} \end{bmatrix}$$

$$= \begin{bmatrix} -1/2^{n-1} \\ -1/2^{n-1} \\ \vdots \\ -1/2^{n-1} \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Moreover, adding $1/2^{n-1}$ to every element in the first column of A gives an exactly singular matrix. Arriving at such a matrix by, say Gaussian elimination, would give no hint as to the near-singularity. However, it is easy to check that $\sigma_n(A)$ (or $1/\|A^{-1}\|$ in any norm) behaves as $1/2^n$. A corollary for control theory: eigenvalues do not necessarily give a reliable measure of "stability margin."

Rank determination, in the presence of roundoff error, is a highly nontrivial problem. And, of course, all the same difficulties arise in any problem equivalent to or involving rank determination such as determining the independence of vectors, finding a basis for ker A, etc. We turn now to some of these problems which naturally arise in linear multivariable control.

IV. Some Applications of SVD

In this section we outline some applications of singular value techniques to the computation of representations for various subspaces and maps that arise in linear algebra and linear systems theory. No proofs will be given nor will the list of topics considered be exhaustive. Rather we shall attempt to impart only the flavor of singular value analysis to the subject, leaving technical matters and the details of software implementation for consideration elsewhere. Notation used will be fairly standard in linear algebra and is consistent with Wonham [31]. We shall consider only the most frequently applied case of real vector spaces and real matrices; however, virtually everything that follows will hold in the complex case as well.

A. Some Basic Properties of Singular Values

Suppose $A\in\mathbb{R}^{m\times n}$ with singular values $\sigma_1\geq\cdots\geq\sigma_n\geq 0$. Then $\|A\|_2=\sigma_1$ and $\|A\|_F=(\sigma_1^2+\cdots+\sigma_n^2)^{1/2}$. Also, if A is invertible, then $\sigma_n>0$ and $\|A^{-1}\|_2=1/\sigma_n$.

If $A\in\mathbb{R}^{n\times n}$ is normal (i.e., $AA^T=A^TA$) with singular values $\sigma_1\geq\cdots\sigma_n\geq 0$, then $\sigma_k=|\lambda_k|$ where $\lambda_k\in\sigma(A)$; $k=1,\cdots,n$. In particular, the singular values and eigenvalues of $A\geq 0$ are identical.

Another fundamental property of singular values, sometimes used as a definition, is the minimax property which is a generalization of the minimax property of eigenvalues of Hermitian matrices. For $A\in\mathbb{R}^{m\times n}$ with singular values $\sigma_1\geq\cdots\geq\sigma_n\geq 0$ we have

$$\sigma_k=\min_{d(\mathcal{S})=n-k+1}\ \max_{\substack{x\in\mathcal{S}\\ x\neq 0}}\frac{\|Ax\|_2}{\|x\|_2};\qquad k=1,\cdots,n.$$

Here $\mathcal{S}$ is a subspace of $\mathbb{R}^n$ and $d(\mathcal{S})$ is the dimension of $\mathcal{S}$.

The minimax property can be used to show easily the well-conditioned nature of singular values. Specifically, if $A\in\mathbb{R}^{m\times n}$ with singular values $\sigma_1\geq\cdots\geq\sigma_n\geq 0$ and if $A+E$ has singular values $\tau_1\geq\cdots\geq\tau_n\geq 0$, then $|\sigma_k-\tau_k|\leq\|E\|_2$; $k=1,\cdots,n$.

Singular values can thus be seen to have intimate connections with the spectral norm. We shall mention more

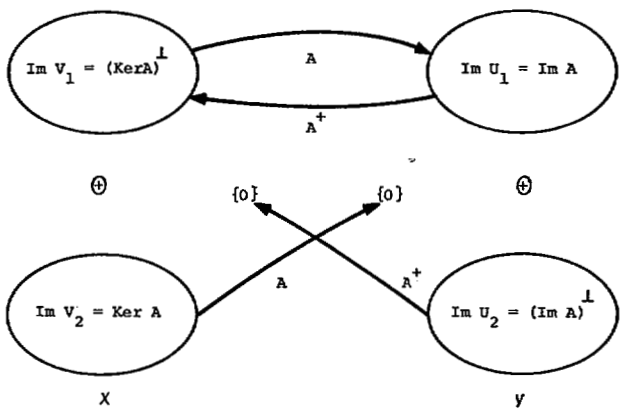
of these relationships in the sequel in the context of solving linear least squares problems.

B. Four Fundamental Subspaces

Consider a linear map $A: X \to Y$ between two finite-dimensional real vector spaces X and Y with $d(X)=n$, $d(Y)=m$. Identifying A with an $m \times n$ real matrix A suppose A has an SVD given by

$$A = U\Sigma V^T = (U_1, U_2)\begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

where $S = \operatorname{diag}(\sigma_1, \cdots, \sigma_r)$ with $\sigma_1 \geqslant \cdots \geqslant \sigma_r > 0$ and U and V are partitioned compatibly (U_1 is $m \times (m-r)$, etc.). Then the U_i and V_i provide orthonormal bases for the four fundamental subspaces in the following self-explanatory diagram:



As discussed in Section II we have the computational problem of intelligently deciding what is σ_r and hence the rank of A. But that decision directly affects each of the above subspaces and hence their calculation. Since SVD is the only generally reliable way of calculating rank it follows that it is the only generally reliable way of calculating bases for these subspaces. Computationally, the orthogonal QR factorization is cheaper and frequently is just as reliable unless one explicitly needs V.

C. Projections

The four fundamental orthogonal projections are given by

$$U_1U_1^T = P_{\operatorname{Im}A} = AA^+$$
$$U_2U_2^T = P_{\ker A^T} = I - AA^+$$
$$V_1V_1^T = P_{\operatorname{Im}A^T} = A^+A$$
$$V_2V_2^T = P_{\ker A} = I - A^+A.$$

For the case of oblique projections, suppose $\operatorname{Im}R = \boldsymbol{R}$ and $\operatorname{Im}S = \boldsymbol{S}$ where $\boldsymbol{R}$ and $\boldsymbol{S}$ are two subspaces of $\boldsymbol{X}$ such that $\boldsymbol{R} \oplus \boldsymbol{S} = \boldsymbol{X}$. Notice that R and S need not have linearly independent columns. Then with the obvious choice of notation we have for the projection of $\boldsymbol{R}$ along $\boldsymbol{S}$

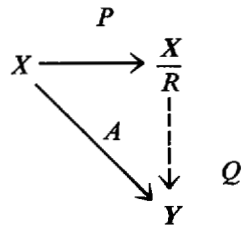
$$P_{\boldsymbol{R},\boldsymbol{S}} = (U_{1R}, 0)(U_{1R}, U_{1S})^{-1}$$

with similar expressions for $P_{\boldsymbol{S},\boldsymbol{R}}$, $P_{\boldsymbol{R},\boldsymbol{S}^\perp}$, etc. Notice that in the special case of $\boldsymbol{S} = \boldsymbol{R}^\perp$ we do indeed have

$$\begin{aligned} P_{\boldsymbol{R},\boldsymbol{R}^\perp} &= (U_{1R}, 0)(U_{1R}, U_{2R})^{-1} \\ &= (U_{1R}, 0)\begin{pmatrix} U_{1R}^T \\ U_{2R}^T \end{pmatrix} \\ &= U_{1R}U_{1R}^T = RR^+ = P_{\operatorname{Im}R} = P_{\boldsymbol{R}}. \end{aligned}$$

D. The Factoring of Linear Map

Suppose $A: \boldsymbol{X} \to \boldsymbol{Y}$, $\boldsymbol{R} \subseteq \ker A \subseteq \boldsymbol{X}$ (here $\subseteq$ denotes "is a subspace of"). Let $P: \boldsymbol{X} \to \boldsymbol{X}/\boldsymbol{R}$ be the canonical projection. Then there exists a unique linear map $Q: \boldsymbol{X}/\boldsymbol{R} \to \boldsymbol{Y}$ such that the following diagram commutes:



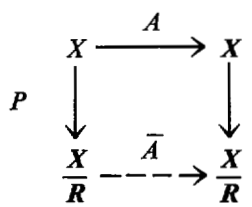
To compute P and Q, suppose $\operatorname{Im}R = \boldsymbol{R}$ and let R have SVD

$$R = (U_1, U_2)\begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}.$$

Then P is given by $P = U_2^T$ while the induced Q is given by $Q = AU_2$. Note that $\ker P = \boldsymbol{R} \subseteq \ker A$ is equivalent to $ARR^+ = 0$ whence $A = A(I - RR^+) = AU_2U_2^T = QP$. In the case when $\boldsymbol{R} = \ker A$ we have $AU_1 = 0$ and $A = QP = (AU_2)$. U_2^T factors A into the product of injective and surjective maps.

E. The Induced Map in the Factor Space

Suppose $A: \boldsymbol{X} \to \boldsymbol{X}$, and $\boldsymbol{R} \subseteq \boldsymbol{X}$ is A-invariant, i.e., $A\boldsymbol{R} \subseteq \boldsymbol{R}$. Then A induces a unique endomorphism $\bar{A}$ of the factor space $\boldsymbol{X}/\boldsymbol{R}$ which makes the following diagram commute:



P = the canonical projection.

Again suppose R is any matrix with $\operatorname{Im}R = \boldsymbol{R}$ and let R have SVD

$$R = (U_1, U_2)\begin{pmatrix} S & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}.$$

Then P is given by U_2^T while $\bar{A}$ is given by $\bar{A}=U_2^TAU_2$. Note that $A\boldsymbol{R}\subseteq\boldsymbol{R}$ is equivalent to

$$RR^+AR=AR$$

which implies $U_2^TAU_1=0$ and $U_2^TA(I-U_2U_2^T)=0$. Thus $\bar{A}P=U_2^TAU_2U_2^T=U_2^TA=PA$. Also note that under the orthogonal change of basis U, A becomes

$$U^TAU=\begin{pmatrix}U_1^TAU_1 & U_1^TAU_2\\ 0 & U_2^TAU_2\end{pmatrix}=\begin{pmatrix}A|\boldsymbol{R} & U_1^TAU_2\\ 0 & \bar{A}\end{pmatrix}.$$

F. Subspaces

Suppose $\boldsymbol{R}\subseteq X$ and $A: X\to X$. Let R be any matrix with $\operatorname{Im}R=\boldsymbol{R}$ and let the SVD of R be as above. Then $d(\boldsymbol{R})=$ the number of nonzero singular values of R and the columns of U_1 give an orthonormal basis for $\boldsymbol{R}$ while the columns of U_2 give an orthonormal basis for $\boldsymbol{R}^\perp\approx(X/\boldsymbol{R})'$ (the prime denotes dual space). A basis for $A\boldsymbol{R}$ can be obtained from the U_1 of the SVD of AR. To get a basis for $A^{-1}\boldsymbol{R}$ we need the U_2 part of the SVD if A^TU_{2R} since $A^{-1}\boldsymbol{R}=(A'\boldsymbol{R}^\perp)^\perp$ where A' is the dual map.

G. The Calculus of Subspaces

Given two subspaces $\boldsymbol{R}$ and $\boldsymbol{T}$ of $\boldsymbol{X}$ it must frequently be determined if $\boldsymbol{R}\subseteq\boldsymbol{T}$. Suppose $\operatorname{Im}R=\boldsymbol{R}$ and $\operatorname{Im}T=\boldsymbol{T}$. Then $\boldsymbol{R}\subseteq\boldsymbol{T}$ if and only if $TT^+R=R$ so using U_1 from the SVD of T one can check if $U_1U_1^TR=R$. Alternatively (but more expensively) one can check if the number of nonzero singular values of T equals the number of nonzero singular values of the matrix $[R,T]$. To check if $\boldsymbol{R}=\boldsymbol{T}$ both $\boldsymbol{R}\subseteq\boldsymbol{T}$ and $\boldsymbol{T}\subseteq\boldsymbol{R}$ can be verified.

Upon computing the SVD of $[R,T]$, the columns of U_1 are an orthonormal basis for $\boldsymbol{R}+\boldsymbol{T}$ while the columns of U_2 are an orthonormal basis for $\boldsymbol{R}^\perp\cap\boldsymbol{T}^\perp$. The columns of V_1 and V_2 do not appear to span anything particularly interesting. The extension of this procedure to arbitrary finite sums of subspaces is obvious.

The "dual calculation" of $\boldsymbol{R}\cap\boldsymbol{T}$ is not quite so easy. One can use $\boldsymbol{R}\cap\boldsymbol{T}=(\boldsymbol{R}^\perp+\boldsymbol{T}^\perp)^\perp$ and do two SVD's to get bases for $\boldsymbol{R}^\perp$, $\boldsymbol{T}^\perp$ then proceed as above with one further SVD. This procedure extends obviously to arbitrary finite intersections of subspaces but is clearly expensive. An alternative procedure (but only for the case of two subspaces) is to do an SVD of $[R,T]$. Then from the partitioning

$$0=[R,T]V_2=[R,T]\begin{pmatrix}V_{2R}\\ V_{2T}\end{pmatrix}$$

choose RV_{2R} or TV_{2T} (according as $d(\boldsymbol{R})$ or $d(\boldsymbol{T})$ is smaller) as the appropriate basis.

H. Angles Between Subspaces

Suppose $\operatorname{Im}R=\boldsymbol{R}\subseteq\boldsymbol{X}$, $\operatorname{Im}S=\boldsymbol{S}\subseteq\boldsymbol{X}$. Let $d(\boldsymbol{R})=r$, $d(\boldsymbol{S})=s$ and suppose, without loss of generality, that $r+s\leqslant n$ and $r\leqslant s$. Then the cosines squared of the r angles of inclination between $\boldsymbol{R}$ and $\boldsymbol{S}$ are given by the r largest eigenvalues of R^+SS^+R. The angles generalize the usual notion of angle when $d(\boldsymbol{X})\leqslant 3$ and are suggestive of a measure of separation between subspaces. If at least one of the angles is 0, $\boldsymbol{R}\cap\boldsymbol{S}\neq 0$. If all angles are nonzero but at least one is "small" then $\boldsymbol{R}\cap\boldsymbol{S}=0$ but we might call the two subspaces "close."

The numerical computation of angles of inclination is treated extensively in [3]. The essential ideas are summarized here and the reader is referred to [3] for details and references. By means of a preliminary QR-factorization or SVD it may be assumed that R, S are orthogonal bases for $\boldsymbol{R}$, $\boldsymbol{S}$. Let $\theta_i=$ the ith angle of inclination; $i=1,\cdots,r$. Then

$$R^+SS^+R=(R^TS)(R^TS)^T$$

so

$$\cos\theta_i=\sigma_i$$

where σ_i is the ith singular value of $M=R^TS$. If M has SVD

$$M=U\Sigma V^T$$

the ith column of RU and the ith column of SV are a pair of vectors in $\boldsymbol{R}$ and $\boldsymbol{S}$, respectively, with angle θ_i between them.

I. Other Applications

There are numerous other important applications of the singular value decomposition when quantitative and qualitative information is desired about a linear map. One of the major areas is in the solution of linear least squares problems. The book by Lawson and Hanson [22] is a good introduction to the subject but the subsequent numerical analysis literature should also be consulted for important recent developments; for example, [28] contains a nice survey of the perturbation theory, and [26] contains a lucid exposition with respect to generalized inverses.

The SVD also finds application in 1) solving systems of linear equations with equality constraints; 2) solving homogeneous systems of linear equations; 3) computing the Moore–Penrose pseudoinverse and other generalized inverses; 4) determining dependencies or near-dependencies among the columns or rows of a matrix (note particularly [16]).

The SVD also finds use as a tool for theoretical analysis of numerical algorithms and we would cite [18] and [5] as examples.

While the SVD plays a useful role in analyzing square, invertible matrices its full power is realized in the analysis of nonsquare, possibly rank-deficient matrices which arise in, for example, linear least squares problems.

We shall treat some of these problems in more detail in the next section in which various computational details associated with using the SVD are discussed.

V. The Use of the Singular Value Decomposition on a Computing Machine

The singular value decomposition of a matrix is one of the most elegant algorithms in numerical algebra for exposing quantitative information about the structure of a system of linear equations.

The condition number of a matrix with respect to the solution of a linear system of equations shows how well the vector x is defined by the transformation $Ax=b$. The condition number $\kappa_2(A)$ of the nonsingular matrix A with singular values $\sigma_1 \geqslant \cdots \geqslant \sigma_n > 0$ is the ratio σ_1/σ_n. For example, if $\kappa(A)=10^6$, a perturbation of 2^{-20} in the elements of A can change the computed solution $\hat{x}$ by a factor of $2^{-20}\cdot 10^6$, that is to say, even the leading digit may be changed. For a more rigorously detailed explanation see [11].

In the discussion that follows, we shall investigate direct computation of the best approximate solution to the possibly overdetermined or underdetermined system of equations $Ax=b$. It is known that reliable information about rank deficiency cannot always be obtained from triangular factorization [16].

The subroutine MINFIT, using the notation in [12], reduces the system of equations

$$Ax=b$$

where A has m rows and n columns (m can be less than, equal to, or greater than n) to the form

$$U\Sigma V^T x=b$$

giving

$$\Sigma V^T x=U^T b.$$

The singular value decomposition can also be computed by the subroutine SVD from [12]. If one needs the explicit columns of U the $m\times m$ identity matrix I_m should be appended to the right-hand side b. There is no restriction, at the subroutine level, on the number of columns of b; the number can be zero.

The diagonal matrix Σ contains the singular values of A. The transformations used to obtain the decomposition preserve unitarily invariant norms, thereby assuring the norm of Σ is that of A. The diagonal elements of Σ, when ordered, are $\sigma_1 \geqslant \sigma_2 \geqslant \sigma_3 \cdots \geqslant \sigma_n \geqslant 0$. MINFIT does not order the singular values. Given information about the certainty of the data A and b, one can choose the best approximating matrix A_r of full rank that is nearest, in the spectral norm sense, to the matrix A. From A_r the best candidate solution x for $Ax=b$ can be computed. If σ_r is chosen such that $\sigma_1 \geqslant \sigma_2 \cdots \geqslant \sigma_r > 0$, $\sigma_{r+1} \geqslant \sigma_{r+2} \cdots \geqslant \sigma_n$, whereby $\sigma_{r+1}, \cdots, \sigma_n$ are effectively considered to be zero, the condition number of A is the ratio σ_1/σ_r. If the matrix A is equilibrated, i.e., scaled, so that $\sigma_1=1$, σ_r should be not less than the square root of the machine precision, or a constant representing the uncertainty in the data, whichever is larger. To be arbitrary about the choice of σ_r relative to σ_1 is difficult. We customarily choose a rank tolerance equal to the floating-point representation of $\epsilon^{1/2}$, the square root of the machine precision. There is an obvious danger that this rank tolerance may be inadequate for some problems. For example, suppose that $A=U\Sigma V^T$ such that

$$\Sigma=\begin{bmatrix} 1 & & & & & & & \\ & 2^{-2} & & & & & & \\ & & 2^{-3} & & & & & \\ & & & 2^{-26} & +\delta & & & \\ & & & & 2^{-26} & & & \\ & & & & & 2^{-27} & & \\ & & & & & & \ddots & \\ & & & & & & & \ddots \end{bmatrix}$$

where $\epsilon \leqslant \delta < 2^{-26}$, say for $\epsilon=2^{-52}$ for IBM 360/370 machines.

The arbitrary rank tolerance would leave σ_4 unchanged but set σ_5 to zero. Thus A_4 would be deemed to have full rank whereas a more judicious choice of rank is 3. This example, though artificial, is given to encourage all users to display the diagonal matrix Σ to see that particular problem's distribution of the singular values.

Given an appropriate choice of σ_r,

$$\|A\|_F - \|A_r\|_F \leqslant \left(\sum_{i=r+1}^{n} (\sigma_i)^2\right)^{1/2}.$$

Noting that $U^TU=V^TV=VV^T=I_n$ and that the pseudoinverse of Σ is the diagonal matrix

$$\Sigma^+=\begin{bmatrix} \operatorname{diag}\left(\frac{1}{\sigma_1}, \cdots, \frac{1}{\sigma_r}\right) & O \\ O & O \end{bmatrix}$$

the pseudoinverse of A is

$$A^+=V\Sigma^+U^T.$$

There is seldom any reason to form a pseudoinverse explicitly. MINFIT accumulates Householder transformations to produce a bidiagonal matrix having the same singular values as A, and continues, by a variant of the QR algorithm (see [17]), to diagonalize the bidiagonal form to give

$$\Sigma V^T x=U^T b=c$$

from which

$$x=V\Sigma^+ c.$$

Various candidate solutions x can be provided by different choices of a rank tolerance to fix σ_r. See [22, chs. 25 and 26] and [19].

For suitably chosen σ_r, consider those columns of V associated with $\sigma_{r+1}, \sigma_{r+2}, \cdots, \sigma_n$ as V_2, namely the columns of V that span ker A. Then

$$AV_2=0.$$

When such columns V_2 exist, they constitute the nontrivial solutions of the homogenous system of equations

$$Ax = 0.$$

The elements of the columns of V can be inspected to reveal dependencies or near dependencies among the columns, i.e., the variables of the coefficient matrix A [16]. Analogously, the columns of U can reveal dependencies among the equations, i.e., the rows of A.

In using MINFIT, we are concerned with three distinct but related items: 1) the stability of the algorithm from the standpoint of numerical algebra, 2) the robustness of the mathematical software that implements the algorithm, and 3) the documentation that provides information on the use of the mathematical software.

The singular value decomposition is stable in the sense that the computation of eigensystems of Hermitian matrices is stable. In general, we expect

$$\frac{\|A - U\Sigma V^T\|}{\|A\|\sqrt{mn}}$$

to be the order of machine precision, as is corroborated for the examples in the sequel. Robustness of this mathematical software is established.

MINFIT can be used to obtain the solution of a linear system of equations. However, if the matrix of coefficients is known to have full rank, and, if the condition number of this matrix is small relative to the uncertainty in the data, one of the matrix factorization methods should be used. Such matrix factorization methods are 1) the Cholesky factorization, 2) the LU decomposition with partial or complete pivoting where the elementary transformations have been stabilized by row and/or column interchanges, and 3) the orthogonal QR factorization with column pivoting [8]. However, such factorizations cannot be *guaranteed* to give definitive information about the condition number of a matrix. Consider from [30] the bidiagonal matrix of order 100

$$\begin{bmatrix} 0.501 & -1 & & & & \\ & 0.502 & -1 & & & \\ & & \ddots & \ddots & & \\ & & & \ddots & \ddots & \\ & & & & 0.599 & -1 \\ & & & & & 0.600 \end{bmatrix}.$$

This matrix is extremely ill-conditioned with respect to the solution of a linear system of equations. Its smallest singular value is approximately 10^{-22} despite the fact that its smallest eigenvalue is 0.501. This matrix also shows that computation of the smallest singular value is limited by the relative finite precision of the machine on which it is computed. That is to say, the small singular value, 10^{-22} will appear computationally to be no smaller than the order of machine precision. This result is not attributable to the construction of the algorithm, but rather to the finite precision of the machine's arithmetic.

We suggest everywhere the use of long precision on the IBM 360/370 machines to compute the solutions of linear systems of equations, eigensystems, and the singular value decomposition. Even so, we urge extreme caution wherever the number of rows m or the number of columns n of a matrix is of more than modest size, say 200, if the matrix is dense. The quantity $\|A - U(\Sigma V^T)\|/\|A\| \cdot \sqrt{mn}$ should be the order of machine precision. However, the computational algorithms are, in general, $0(n^3)$ or $0(mn \cdot \min\{m,n\})$ processes. We advise a rigorous analysis of the structure of the matrix of high dimensions before any of the numerical algebra algorithms are used.

The singular values of a matrix can be substantially altered by scaling the original data matrix as is shown by the examples that follow. Deliberately, MINFIT does not include scaling of the rows or columns of the matrix A or right-hand sides b. For the best performance of the algorithm we suggest that columns of A be equilibrated such that the sums of their elements be as nearly equal as possible. Exact powers of 16 for IBM 360/370 machines should be used for scaling factors so that the data is not perturbed in trailing digits. Row scaling will have the effect of introducing weights on the data if one has a least squares problem and therefore should be done at a user's discretion. An excellent discussion of scaling is in [22].

Lawson further points out in [22] that it is important to take advantage of information about the certainty of data. For example, if data are known to have uncertainty in the third decimal place, that digit and all that follow are arbitrary. The matrix

$$\begin{bmatrix} 1.02 & 1.09 \\ 1.05 & 1.01 \end{bmatrix}$$

if uncertain in the third figure could lead to

$$\begin{bmatrix} 1.00 & 1.00 \\ 1.00 & 1.00 \end{bmatrix}.$$

The eigenvectors of a symmetric matrix, and therefore, the singular vectors U and V from MINFIT are known only to within a constant multiplier of modulus 1. If anyone should attempt to recompute the results that follow on a machine whose arithmetic is different from that of the IBM 370/168 a change in sign on the columns of U or V may be observed.

VI. Conclusions

We have given a descriptive introduction to the singular value decomposition from the point of view of its potential applications in linear systems theory. A number of these applications have been explicitly exhibited. There is, throughout the paper, an emphasis on certain important details of the implementation of SVD on a digital computer. A number of illustrative examples and computed solutions are included for reference and verification purposes in an Appendix.

It suffices to say, in conclusion, that singular value analysis forms a cornerstone of modern numerical linear

algebra and we are now just beginning to see its legitimate use in linear control and systems theory.

APPENDIX
SELECTED MATRICES, COMPUTED SOLUTIONS, AND ILLUSTRATIVE EXAMPLES

We display a representative sample of matrices on which the subroutine MINFIT has performed satisfactorily. The format of the printing was chosen for convenience and does not include the full fifteen decimal place output that was produced by the long precision computation on the machine. If anyone should attempt to reproduce these results on a machine whose arithmetic or relative precision is different from that of the IBM 370/168, output may be obtained that is different from that which we display. However, such results should be correct to the order of machine precision on which the computation is performed.

Although we include certain matrices of the Hilbert segments, we do not encourage their use as test matrices for software validation. The Hilbert segments are not representable exactly in a computing machine unless appropriate multipliers are used to preclude a perturbation on input of the data. We have used such multipliers.

Other matrices exhibited are a 3×3 matrix that is contrived to display information about near dependencies of rows or columns and a test matrix from [4]. The matrix from [4] is exactly representable in the machine though it is ill-conditioned with respect to the solution of linear systems of equations.

On the output that is displayed, V has its usual meaning, W contains the unordered singular values from MINFIT, P is an integer vector that indicates the descending order of the singular values, and C contains U^Tb. X contains the candidate solution of $Ax=b$. IERR is the error indicator from MINFIT; it is nonzero if the computation of any singular value requires more than 30 iterations or if the maximum singular value is zero.

Example 1

This 3×3 matrix shows output that indicates rank 2 if the smallest singular value is treated as zero. Given this interpretation, columns 1 and 2 are linearly dependent. This information is contained in column 2 of the V matrix.

```
 A=
( ROW    1 ):
0.10101000 01  0.10098000 01  0.98000000 00
( ROW    2 ):
0.10098000 01  0.10104000 01  0.98000000 00
( ROW    3 ):
0.98000000 00  0.98000000 00  0.10100000 01


 B=
(COLUMN   1)
 0.10000000 01  0.0            0.0
(COLUMN   2)
 0.0            0.10000000 01  0.0
(COLUMN   3)
 0.0            0.0            0.10000000 01


IERR =    0

 V =
(COLUMN   1)
-0.57927490 00 -0.57933300 00 -0.57342300 00
(COLUMN   2)
-0.70801190 00  0.70619830 00  0.17604610-02
(COLUMN   3)
-0.40393050 00 -0.40701010 00  0.81925760 00


 W=
 0.29901010 01  0.44980760-03  0.39948830-01


 C=
(COLUMN   1)
-0.57927490 00 -0.70801190 00 -0.40393050 00
(COLUMN   2)
-0.57933300 00  0.70619830 00 -0.40701010 00
(COLUMN   3)
-0.57342300 00  0.17604610-02  0.81925760 00

 P =     1    3    2


USING MACHEP, X=
(COLUMN   1)
 0.11186300 04 -0.11073520 04 -0.10943610 02
(COLUMN   2)
-0.11073520 04  0.11129910 04 -0.54718030 01
(COLUMN   3)
-0.10943610 02 -0.54718030 01  0.16917920 02


 USING RKTOL, X=
(COLUMN   1)
 0.11186300 04 -0.11073520 04 -0.10943610 02
(COLUMN   2)
-0.11073520 04  0.11129910 04 -0.54718030 01
(COLUMN   3)
-0.10943610 02 -0.54718030 01  0.16917920 02
```

Example 2

The Hilbert matrix of order 7, generated in long precision, 7 digits of which are given for each element, is *inexact in the machine.*

```
 A=
( ROW    1 ):
0.10000000 01  0.50000000 00  0.33333330 00  0.25000000 00  0.20000000 00  0.16666670 00  0.14285710 00
( ROW    2 ):
0.50000000 00  0.33333330 00  0.25000000 00  0.20000000 00  0.16666670 00  0.14285710 00  0.12500000 00
( ROW    3 ):
0.33333330 00  0.25000000 00  0.20000000 00  0.16666670 00  0.14285710 00  0.12500000 00  0.11111110 00
( ROW    4 ):
0.25000000 00  0.20000000 00  0.16666670 00  0.14285710 00  0.12500000 00  0.11111110 00  0.10000000 00
( ROW    5 ):
0.20000000 00  0.16666670 00  0.14285710 00  0.12500000 00  0.11111110 00  0.10000000 00  0.90909090-01
( ROW    6 ):
0.16666670 00  0.14285710 00  0.12500000 00  0.11111110 00  0.10000000 00  0.90909090-01  0.83333330-01
( ROW    7 ):
0.14285710 00  0.12500000 00  0.11111110 00  0.10000000 00  0.90909090-01  0.83333330-01  0.76923080-01
```

Its singular values are

```
W=
0.16608850 01  0.2719202D 00  0.2128975D-01  0.10085880-02  0.2938637D-04  0.4856763D-06  0.3493744D-08
```

Example 3

Multiplication of the Hilbert matrix of order 7 by the constant 360360 allows a machine representation that is exact.

```
 A=
( ROW     1 ):
0.3603600D 06  0.1801800D 06  0.1201200D 06  0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05
( ROW     2 ):
0.1801800D 06  0.1201200D 06  0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05
( ROW     3 ):
0.1201200D 06  0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05
( ROW     4 ):
0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05
( ROW     5 ):
0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05  0.3276000D 05
( ROW     6 ):
0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05  0.3276000D 05  0.3003000D 05
( ROW     7 ):
0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05  0.3276000D 05  0.3003000D 05  0.2772000D 05
```

Its singular values are

```
W=
0.5985166D 06  0.9798916D 05  0.7671976D 04  0.3634546D 03  0.1058967D 02  0.1750183D 00  0.1259061D-02
```

Example 4

The Bauer matrix [4] with its associated output is

```
  A=
 ( ROW     1 ):
-0.7400000D 02  0.8000000D 02  0.1800000D 02 -0.1100000D 02 -0.4000000D 01 -0.8000000D 01
 ( ROW     2 ):
 0.1400000D 02 -0.6900000D 02  0.2100000D 02  0.2800000D 02  0.0             0.7000000D 01
 ( ROW     3 ):
 0.6600000D 02 -0.7200000D 02 -0.5000000D 01  0.7000000D 01  0.1000000D 01  0.4000000D 01
 ( ROW     4 ):
-0.1200000D 02  0.6600000D 02 -0.3000000D 02 -0.2300000D 02  0.3000000D 01 -0.3000000D 01
 ( ROW     5 ):
 0.3000000D 01  0.8000000D 01 -0.7000000D 01 -0.4000000D 01  0.1000000D 01  0.0
 ( ROW     6 ):
 0.4000000D 01 -0.1200000D 02  0.4000000D 01  0.4000000D 01  0.0             0.1000000D 01


 B=
(COLUMN    1)
 0.5100000D 02 -0.6100000D 02 -0.5600000D 02  0.6900000D 02  0.1000000D 02 -0.1200000D 02
(COLUMN    2)
-0.5600000D 02  0.5200000D 02  0.7640000D 03  0.4096000D 04 -0.1327600D 05  0.8421000D 04
(COLUMN    3)
-0.5000000D 01 -0.9000000D 01  0.7080000D 03  0.4165000D 04 -0.1326600D 05  0.8409000D 04


IERR =     0

 V =
(COLUMN    1)
 0.5315959D 00 -0.8242984D 00  0.3824286D-01  0.1794925D 00  0.1057691D-01  0.6439019D-01
(COLUMN    2)
-0.6250950D 00 -0.2981574D 00  0.6284508D 00  0.3481670D 00 -0.6385690D-01  0.1049137D-01
(COLUMN    3)
 0.3369620D 00  0.1042175D 00  0.6565848D 00 -0.5238190D 00 -0.2350173D 00 -0.3389280D 00
(COLUMN    4)
-0.4082483D 00 -0.4082483D 00 -0.4082483D 00 -0.4082483D 00 -0.4082483D 00 -0.4082483D 00
(COLUMN    5)
 0.2153923D 00  0.2325174D 00 -0.7045919D-01  0.6062083D 00 -0.6389627D 00 -0.3446961D 00
(COLUMN    6)
 0.7629926D-02 -0.6490533D-02 -0.2919267D-01  0.1949887D 00  0.6046795D 00 -0.7716149D 00


 W=
 0.1738393D 03  0.6486187D 02  0.1066716D 02  0.1000000D 01  0.1752477D 00  0.4744182D-04
```

Example 5

The condition number of a nonsingular matrix may be improved by row or column scaling. The Bauer matrix, scaled as

```
 A=
 ( ROW    1 ):
-0.7400000D 02  0.8000000D 02  0.3600000D 02 -0.3300000D 02 -0.4000000D 02 -0.8000000D 02
 ( ROW    2 ):
 0.1400000D 02 -0.6900000D 02  0.4200000D 02  0.8400000D 02  0.0            0.7000000D 02
 ( ROW    3 ):
 0.6600000D 02 -0.7200000D 02 -0.1000000D 02  0.2100000D 02  0.1000000D 02  0.4000000D 02
 ( ROW    4 ):
-0.1200000D 02  0.6600000D 02 -0.6000000D 02 -0.6900000D 02  0.3000000D 02 -0.3000000D 02
 ( ROW    5 ):
 0.2400000D 02  0.6400000D 02 -0.1120000D 03 -0.9600000D 02  0.8000000D 02  0.0
 ( ROW    6 ):
 0.2800000D 02 -0.8400000D 02  0.5600000D 02  0.8400000D 02  0.0            0.7000000D 02
```

has singular values

```
W=
0.2959449D 03  0.1816570D 03  0.4893780D 02  0.1288217D 02  0.7095995D 00  0.1397107D-02
```

Example 6

The singular value decomposition provides $U\Sigma V^T$ as the decomposition of a matrix A. Given the orthonormal columns U and V one can form another matrix $U\Sigma V^T$ for arbitrary Σ. Using U and V from the inexact Hilbert matrix of order 7, the reformed matrix

```
 THE REFORMED A =
 ( ROW    1 ):
 0.2010649D 02 -0.1119123D 03  0.2325041D 03 -0.2489384D 03  0.1477433D 03 -0.4637337D 02  0.6040208D 01
 ( ROW    2 ):
-0.1119123D 03  0.1320994D 04 -0.4728001D 04  0.7849842D 04 -0.6765319D 04  0.2949071D 04 -0.5155581D 03
 ( ROW    3 ):
 0.2325041D 03 -0.4728001D 04  0.2536401D 05 -0.5888948D 05  0.6799662D 05 -0.3855889D 05  0.8582491D 04
 ( ROW    4 ):
-0.2489384D 03  0.7849842D 04 -0.5888948D 05  0.1800569D 06 -0.2636029D 06  0.1847044D 06 -0.4986868D 05
 ( ROW    5 ):
 0.1477433D 03 -0.6765319D 04  0.6799662D 05 -0.2636029D 06  0.4713941D 06 -0.3930209D 06  0.1238620D 06
 ( ROW    6 ):
-0.4637337D 02  0.2949071D 04 -0.3855889D 05  0.1847044D 06 -0.3930209D 06  0.3792650D 06 -0.1355068D 06
 ( ROW    7 ):
 0.6040208D 01 -0.5155581D 03  0.8582491D 04 -0.4986868D 05  0.1238620D 06 -0.1355068D 06  0.5368992D 05
```

where σ_i are 10^6, 10^5, 10^4, 10^3, 10^2, 10^1, and 10^0.

The computed σ_i from the reformed A are

```
 W=
 0.1000000D 07  0.1000000D 06  0.1000000D 05  0.1000000D 04  0.1000000D 03  0.1000000D 01  0.1000000D 02

MAX-ROW-SUM RESIDUAL =     0.1228389490D-14
EUCLIDEAN RESIDUAL =     0.9990491258D-15
MAX-COL-SUM RESIDUAL =     0.1228389490D-14
```

However, choosing $\sigma_i = 10^{24}$, 10^{20}, 10^{16}, 10^{12}, 10^8, 10^4, 10^0 gives

```
 W=
 0.1000000D 25  0.1000000D 21  0.1000000D 17  0.2608234D 08  0.1000001D 13  0.2189502D 08  0.3546465D 07

MAX-ROW-SUM RESIDUAL =     0.6067562603D-15
EUCLIDEAN RESIDUAL =     0.4697620457D-15
MAX-COL-SUM RESIDUAL =     0.4045041735D-15
```

The singular values smaller than 10^{12} are affected by the order of machine precision relative to the maximum singular value.

Choosing $\sigma_i = 10^0$, 10^{-4}, 10^{-8}, 10^{-12}, 10^{-16}, 10^{-20}, 10^{-24} gives

```
 W=
 0.1000000D 01  0.1000000D-03  0.1000000D-07  0.9999812D-12  0.1104082D-15  0.1852770D-16  0.2530714D-17

MAX-ROW-SUM RESIDUAL =     0.1316051070D-14
EUCLIDEAN RESIDUAL =     0.8881784153D-15
MAX-COL-SUM RESIDUAL =     0.2632102139D-15
```

Example 7

The order 100 matrix [30]

$$\begin{bmatrix} 0.501 & -1 & & & & \\ & 0.502 & -1 & & & \\ & & \cdot & \cdot & & \\ & & & \cdot & \cdot & \\ & & & & \cdot & -1 \\ & & & & & 0.600 \end{bmatrix}$$

has a maximum singular value ~ 1.587 and a minimum singular value $\sim 10^{-22}$. The minimum singular value computed on the IBM 370/168 is $0.3329410 \times 10^{-15}$. Using long precision on the IBM 370/195 at Argonne National Laboratory. J. Dongarra computed the same singular values as those from the 168 except for the minimum singular value which was $0.33292721 \times 10^{-15}$. The arithmetic of the 195 is not the same as that of the 168. Multiplying this matrix by 10^3 (so that the input was internally representable as exact integers) gave the smallest singular value $0.33294095 \times 10^{-12}$. B. Smith suggested running this matrix on the 195 using short precision from which the smallest singular value was 0.1287991×10^{-5} and $0.13423073 \times 10^{-2}$ for the matrix scaled by 10^3.

Timing tests have been done on the singular value decomposition (SVD and MINFIT). Some reported times on a variety of machines with a variety of compilers are published in [12].

REFERENCES

[1] T. J. Aird, *The Fortran Converter User's Guide*, IMSL, 1975.
[2] L. Autonne, *Bull. Soc. Math. France*, vol. 30, pp. 121–133, 1902.
[3] A. Bjorck and G. H. Golub, "Numerical methods for computing angles between linear subspaces," *Math. Comput.*, vol. 27, pp. 579–594, 1973.
[4] F. L. Bauer, "Elimination with weighted row combinations for solving linear equations and least squares problems," in *Handbook for Automatic Computation*, vol. II: *Linear Algebra*, J. H. Wilkinson and C. Reinsch, Eds. Berlin: Springer-Verlag, 1971, pp. 338–352.
[5] A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson, "An estimate for the condition number of a matrix," Argonne Nat. Lab., Appl. Math. Div., LINPACK Working Note 7, TM-310, July 1977.
[6] W. Cowell, *Portability of Numerical Software* (Oak Brook, IL), *Lecture Notes in Computer Science*, vol. 57. New York: Springer-Verlag, 1977.
[7] G. Dahlquist and A. Bjorck, *Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1974.
[8] J. J. Dongarra *et al.*, *LINPACK User's Guide*. Philadelphia, PA: SIAM, 1979.
[9] C. Eckart and G. Young, "A principal axis transformation for non-Hermitian matrices," *Bull. Amer. Math. Soc.*, vol. 45, pp. 118–121, 1939.
[10] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer Methods for Mathematical Computations*. Englewood Cliffs, NJ: Prentice-Hall, 1977.
[11] G. E. Forsythe and C. B. Moler, *Computer Solution of Linear Algebraic Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1967.
[12] B. S. Garbow *et al.*, *Matrix Eigensystem Routines—EISPACK Guide Extension* (*Lect. Notes in Comput. Sci.*), vol. 51. New York: Springer-Verlag, 1977.
[13] W. Givens, "Numerical computation of the characteristic values of a real symmetric matrix," ORNL Rep. 1574, 1954.
[14] I. C. Gohberg and M. G. Krein, *Introduction to the Theory of Linear Non-Self-Adjoint Operators*. Providence, RI: Amer. Math. Soc., 1969.
[15] G. H. Golub and W. Kahan, "Calculating the singular values and pseudo-inverse of a matrix," *SIAM J. Numer. Anal.*, vol. 2, pp. 205–224, 1965.
[16] G. H. Golub, V. C. Klema, and G. W. Stewart, "Rank degeneracy and least squares problems," Dep. Comput. Sci., Stanford, Univ. Tech. Rep. STAN-CS-76-559, Aug. 1976.
[17] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numer. Math.*, vol. 14, pp. 403–420, 1970.
[18] G. H. Golub and J. H. Wilkinson, "Ill-conditioned eigensystems and the computation of the Jordan canonical form," *SIAM Rev.*, vol. 18, pp. 578–619, 1976.
[19] R. J. Hanson and C. L. Lawson, "Extensions and applications of the Householder algorithm for solving linear least squares problems," *Math. Comput.*, vol. 23, pp. 787–812, 1969.
[20] W. Kahan, private communication.
[21] ——, "Numerical linear algebra," *Can. Math. Bull.*, vol. 9, pp. 756–801, 1966.
[22] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*. Englewood Cliffs, NJ: Prentice-Hall, 1974.
[23] C. C. MacDuffee, *The Theory of Matrices*. Berlin: Springer, 1933.
[24] A. M. Ostrowski, "On the spectrum of a one-parametric family of matrices, *J. Reine Angew. Math.*, vol. 193, pp. 143–160, 1954.
[25] B. T. Smith *et al.*, *Matrix Eigensystem Routines—EISPACK Guide*, 2nd ed. (*Lect. Notes in Comput. Sci.*) vol. 6. New York: Springer-Verlag, 1976.
[26] T. Soderstrom and G. W. Stewart, "On the numerical properties of an iterative method for computing the Moore-Penrose generalized inverse," *SIAM J. Numer. Anal.*, vol. 11, no. 1, pp. 61–74, 1974.
[27] G. W. Stewart, *Introduction to Matrix Computations*. New York: Academic, 1973.
[28] ——, "On the perturbation of pseudo-inverses, projections, and linear least squares problems," *SIAM Rev.*, vol. 19, pp. 634–662, 1977.
[29] J. H. Wilkinson, *Rounding Errors in Algebraic Processes*. Englewood Cliffs, NJ: Prentice-Hall, 1963.
[30] ——, *The Algebraic Eigenvalue Problem*. London, England: Oxford Univ. Press, 1965.
[31] W. M. Wonham, *Linear Multivariable Control: A Geometric Approach*. New York: Springer-Verlag, 1974.
[32] ——, "Geometric state-space theory in linear multivariable control: A status report," *Automatica*, vol. 15, pp. 5–13, 1979.

Virginia C. Klema received the A. B. degree in mathematics from LaGrange College, LaGrange, GA, in 1949 and the M.S. degree in mathematics from Auburn University, Auburn, AL, in 1951.

She worked as an Associate Mathematician in the Mathematics Division, Oak Ridge National Laboratory. Subsequently she worked as a lecturer in the Department of Mathematics, Northwestern University, and as a research staff member, Applied Mathematics Division, Argonne National Laboratory. She was a Senior Research Associate, NBER Computer Research Center, from 1972–1978. Presently she is a member of the research staff, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge. She serves an an Associate Editor for *Transactions on Mathematical Software*. Her current research interests are robust mathematical software for numerical linear algebra and its applications to statistics, optimization, and control.

Alan J. Laub (M'75) received the B.Sc. degree in mathematics from the University of British Columbia, Vancouver, B.C., Canada, in 1969, and the M.S. degree in mathematics and the Ph.D. degree in control sciences both from the University of Minnesota, Minneapolis, in 1972 and 1974, respectively.

From 1974–1975 he was an Assistant Professor in the Department of Systems Engineering, Case Western Reserve University; from 1975–1977 he was a visiting Assistant Professor in the Department of Electrical Engineering, University of Toronto; and from 1977–1979 he was on the research staff of the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology. He is presently Associate Professor of Electrical Engineering at the University of Southern California, and serves as an Associate Editor of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL. His current research interests are in numerical analysis, mathematical software, and large-scale systems theory.