Numerical Linear Algebra Techniques for Systems and Control

Introduction and Survey of IEEE Reprints book

R.V. Patel A.J. Laub P.M. Van Dooren

January 18, 1993

1 Introduction

This part provides an introduction to various aspects of the numerical solution of selected problems of interest in systems, control, and estimation theory. Space limitations preclude an exhaustive survey; rather, a compact "introduction to the literature" will lead the interested reader to sources of additional detailed information and help to put the other parts in this reprints book in proper perspective.

Many of the problems considered in this book arise in the study of the "standard" linear model

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) && (1) \\ y(t) &= Cx(t) + Du(t). && (2) \end{aligned}$$

Here, $x(t)$ is an n-vector of states, $u(t)$ is an m-vector of controls or inputs, and $y(t)$ is an p-vector of outputs. The standard discrete-time analog of (1), (2) takes the form

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k && (3) \\ y_k &= Cx_k + Du_k. && (4) \end{aligned}$$

Of course, considerably more elaborate models are also studied, including time-varying, stochastic, and nonlinear versions of the above, but these will not be discussed in this book. In fact, the above linear models are usually derived from linearizations of nonlinear models about selected nominal points. The interested reader is referred to standard textbooks such as [2], [17], [109], and [201], for further details.

The matrices considered here will, for the most part, be assumed to have real coefficients and be small (of order a few hundred or less) and dense, with no particular exploitable structure. Calculations for most problems in classical single-input, single-output control fall into this category. It must be emphasized that consideration of large, sparse matrices or matrices with special, exploitable structures may involve significantly different concerns and methodologies than those to be discussed here.

The systems, control, and estimation literature is replete with *ad hoc* algorithms to solve the computational problems which arise in the various methodologies. Many of these algorithms work quite well on some problems (e.g., "small order" matrices) but encounter numerical difficulties, often severe, when "pushed" (e.g., on larger order matrices). The reason for this is that little or no attention has been paid to how the algorithms will perform in "finite arithmetic," i.e., on a finite-word-length digital computer.

A simple example due to Moler and Van Loan [144] will illustrate a typical pitfall. Suppose it is desired to compute the matrix e^A in single precision arithmetic on a computer which gives 6 decimal places of precision in the fraction part of floating-point numbers. Consider the case

$$A = \begin{bmatrix} -49 & 24 \\ -64 & 31 \end{bmatrix}$$

and suppose the computation is attempted using the Taylor series formula

$$e^A = \sum_{k=0}^{+\infty} \frac{1}{k!} A^k. \qquad (5)$$

This is easily coded and it is determined that the first 60 terms in the series suffice for the computation, in the sense that terms for $k \geq 60$ are of the order of 10^{-7} and no longer add anything significant to the sum. The resulting answer is

$$\begin{bmatrix} -22.2588 & -1.43277 \\ -61.4993 & -3.47428 \end{bmatrix}.$$

Unfortunately, the true answer is (correctly rounded)

$$\begin{bmatrix} -0.735759 & 0.551819 \\ -1.47152 & 1.10364 \end{bmatrix}$$

and one sees a rather alarming disparity. What happened here was that the intermediate terms in the series got very large before the factorial began to dominate. In fact, the 17th and 18th terms, for example, are of the order of 10^7 but of opposite signs so that the less significant parts of these numbers—while significant for the final answer—are "lost" because of the finiteness of the arithmetic.

Now for this particular example various fixes and remedies are available. But in more realistic examples one seldom has the luxury of having the "true answer" available so that it is not always easy simply to inspect or test an answer such as the one obtained above and determine it to be in error. Mathematical analysis (truncation of the series, in the example above) alone is simply not sufficient when a problem is analyzed or solved in finite arithmetic (truncation of the arithmetic). Clearly, a great deal of care must be taken.

The finiteness inherent in representing real or complex numbers as floating-point numbers on a digital computer manifests itself in two important ways: floating-point numbers have only finite precision and finite range. In fact, it is the degree of attention paid to these two considerations that distinguishes many reliable algorithms from more unreliable counterparts. Wilkinson [197] still provides the definitive introduction to the vagaries of floating-point computation while [133] and the references therein may be consulted to bring the interested reader up to date on roundoff analysis.

The development in systems, control, and estimation theory, of stable, efficient, and reliable algorithms which respect the constraints of finite arithmetic began in the 1970's and is ongoing. Much of the research in numerical analysis has been directly applicable, but there are many computational issues in control (e.g., the presence of hard or structural zeros) where numerical analysis does not provide a ready answer or guide. A symbiotic relationship has developed, particularly between numerical linear algebra and linear system and control theory, which is sure to provide a continuing source of challenging research areas.

The abundance of numerically fragile algorithms is partly explained by the following observation which will be emphasized by calling it a "folk theorem":

> If an algorithm is amenable to "easy" hand calculation, it is probably a poor method if implemented in the finite floating-point arithmetic of a digital computer.

For example, when confronted with finding the eigenvalues of a 2×2 matrix most people would find the characteristic polynomial and solve the resulting quadratic equation. But when extrapolated as a general method for computing eigenvalues and implemented on a digital computer, this turns out to be a very poor procedure indeed for a variety of reasons (such as roundoff and overflow/underflow). Of course the preferred method now would generally be the double Francis

QR algorithm (see [60], [61], [171], and [198] for the messy details) but few would attempt that by hand—even for very small order problems.

In fact, it turns out that many algorithms which are now considered fairly reliable in the context of finite arithmetic are not amenable to hand calculations (e.g., various classes of orthogonal similarities). This is sort of a converse to the folk theorem. Particularly in linear system and control theory, we have been too easily seduced by the ready availability of closed-form solutions and numerically naive methods to implement those solutions. For example, in solving the initial value problem

$$\dot{x}(t) = Ax(t); \qquad x(0) = x_0 \tag{6}$$

it is not at all clear that one should explicitly want to compute the intermediate quantity e^{tA}. Rather, it is the vector $e^{tA}x_0$ that is desired, a quantity that may be computed more reasonably by treating (6) as a system of (possibly stiff) differential equations and using, say, an implicit method for numerical integration of the differential equation. But such techniques are definitely not attractive for hand computation.

Awareness of such numerical issues in the mathematics and engineering community has increased significantly in the last fifteen years or so. In fact, some of the background material (mentioned in this part and in some of the papers in later parts) that is well known to numerical analysts, has already filtered down to undergraduate and graduate curricula in these disciplines. A number of introductory textbooks currently available (e.g., [90], [34], [56], [85], [164], [165]) also reflect a strong software component. The effect of this awareness and education has been particularly noticeable in the area of system and control theory, especially in linear system theory. A number of numerical analysts were attracted by the wealth of interesting numerical linear algebra problems in linear system theory. At the same time, several researchers in the area of linear system theory turned their attention to various methods and concepts from numerical linear algebra and attempted to modify and use them in developing reliable algorithms and software for specific problems in linear system theory. This cross-fertilization has been greatly enhanced by the widespread use of software packages and by recent developments in numerical linear algebra. This process has already begun to have a significant impact on the future directions and development of system and control theory, and applications, as is evident from the growth of computer-aided control system design (CACSD) as an intrinsic tool. Algorithms implemented as mathematical software are a critical "inner" component of a CACSD system.

In the remainder of this part, we give a survey of some recent results and trends in this interdisciplinary research area. We put the emphasis on numerical aspects of the problems/algorithms, which is why we also spend some time in going over the appropriate numerical tools and techniques. We discuss a number of control and filtering problems that are of widespread interest to the control systems community.

Before proceeding further we shall list here some notation to be used in the sequel:

$\mathbb{F}^{n\times m}$	the set of all $n \times m$ matrices with coefficients in the field $\mathbb{F}$ ($\mathbb{F}$ will generally be $\mathbb{R}$ or $\mathbb{C}$)
A^T	the transpose of $A \in \mathbb{R}^{n\times m}$
A^H	the complex-conjugate transpose of $A \in \mathbb{C}^{n\times m}$
A^+	the Moore-Penrose pseudoinverse of A
$\|A\|$	the spectral norm of A (i.e., the matrix norm subordinate to the Euclidean vector norm: $\|A\| = \max_{\|x\|_2=1} \|Ax\|_2$)
diag $(a_1, \cdots, a_n)$	the diagonal matrix $\begin{bmatrix} a_1 & & 0 \\ & \ddots & \\ 0 & & a_n \end{bmatrix}$
$\Lambda(A)$	the set of eigenvalues $\lambda_1, \cdots, \lambda_n$ (not necessarily distinct) of $A \in \mathbb{F}^{n\times n}$
$\lambda_i(A)$	the ith eigenvalue of A
$\Sigma(A)$	the set of singular values $\sigma_1, \cdots, \sigma_m$ (not necessarily distinct) of $A \in \mathbb{F}^{n\times m}$
$\sigma_i(A)$	the ith singular value of A.

Finally, let us define a particular number to which we shall make frequent reference in the sequel. The *machine epsilon* or *relative machine precision* can be defined, roughly speaking, as the smallest positive number ϵ which, when added to 1 on our computing machine, gives a number greater than 1. In other words, any machine representable number δ less than ϵ gets " rounded off" when (floating-point) added to 1 to give exactly 1 again as the rounded sum. The number ϵ, of course, varies depending on the kind of computer being used and the precision with which the computations are being done (single precision, double precision, etc.). But the fact that there exists such a positive number ϵ is entirely a consequence of finite word length.

2 Numerical Background

In this section we give a very brief discussion of two concepts of fundamental importance in numerical analysis: *numerical stability* and *conditioning*. While this material is standard in textbooks such as [73],[39],[79],[174], [179] it is presented here both for completeness and because the two concepts are frequently confused in the systems, control, and estimation literature.

Suppose we have some mathematically defined problem represented by f which acts on data d belonging to some set of data $\mathcal{D}$, to produce a solution $f(d)$ in a solution set $\mathcal{S}$. These notions are kept deliberately vague for expository purposes. Given $d \in \mathcal{D}$ we desire to compute $f(d)$. Suppose d^* is some approximation to d. If $f(d^*)$ is "near" $f(d)$ the problem is said to be well-conditioned. If $f(d^*)$ may potentially differ greatly from $f(d)$ even when d^* is near d, the problem is said to be ill-conditioned. The concept of "near" can be made precise by introducing norms in the appropriate spaces. We can then define the condition of the problem f with respect to these norms as

$$\kappa[f(d)] = \lim_{\delta\to 0} \sup_{d_2(d,d^*)=\delta} \left[\frac{d_1(f(d), f(d^*))}{\delta} \right] \tag{7}$$

where d_i (. , .) are distance functions in the appropriate spaces. When $\kappa[f(d)]$ is infinite, the problem of determining *f(d)* from *d* is *ill-posed* (as opposed to *well-posed*). When $\kappa[f(d)]$ is finite and *relatively large* (or *relatively small*), the problem is said to be *ill-conditioned* (or *well-conditioned*). Further details can be found in [163].

A simple example of an ill-conditioned problem is the following. Consider the $n \times n$ matrix

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

with n eigenvalues at 0. Now consider a small perturbation of the data (the n^2 elements of A) consisting of adding the number 2^{-n} to the first element in the last (nth) row of A. This perturbed matrix then has n distinct eigenvalues $\lambda_1, \cdots, \lambda_n$ with $\lambda_k = 1/2 \exp(2k\pi j/n)$. Thus, we see that this small perturbation in the data has been magnified by a factor on the order of 2^n to result in a rather large perturbation in the solution (the eigenvalues of A). Further details and related examples can be found in [198].

Note that we have thus far made no mention of how the problem f above (computing $\Lambda(A)$ in the example) was to be solved. Conditioning is a function solely of the problem itself. To solve a problem numerically we typically must implement some numerical procedure or algorithm which we shall denote by f^*. Thus, given d, $f^*(d)$ represents the result of applying the algorithm to d (for simplicity, we assume d is "representable"; a more general definition can be given when some approximation d^{**} to d must be used). The algorithm f^* is said to be numerically (backward) stable if, for all $d \in \mathcal{D}$, there exists $d^* \in \mathcal{D}$ near d such that $f^*(d)$ is near $f(d^*)$ (= the exact solution of a nearby problem). If the problem is well-conditioned, then $f(d^*)$ will be near $f(d)$ so that $f^*(d)$ will be near $f(d)$ if f^* is numerically stable. In other words, f^* does not introduce any more sensitivity to perturbation than is inherent in the problem. Example 1 below will further illuminate this definition of stability which, on a first reading, can seem somewhat confusing.

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. Example 2, below, will illustrate this phenomenon. 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 $f(d^*)$, for some d^*, and second, if the problem is well-conditioned then, as above, $f(d^*)$ is near $f(d)$. Thus, $f^*(d)$ is near $f(d)$ and we have an "accurate" solution.

Roundoff errors can cause unstable algorithms to give disastrous results. However, it would be virtually impossible to account for every roundoff error made at every arithmetic operation in a complex series of calculations such as those involved in most linear algebra calculations. This would constitute a *forward* error analysis. The concept of *backward* error analysis based on the definition of numerical stability given above provides a more practical alternative. To illustrate this, let us consider the singular value decomposition of an arbitrary $m \times n$ matrix A with coefficients in $\mathbb{R}$ or $\mathbb{C}$ [73]

$$A = U \Sigma V^H \ . \tag{8}$$

Here U and V are $m \times m$ and $n \times n$ unitary matrices, respectively, and Σ is an $m \times n$ matrix of the form

$$\Sigma = \left[\begin{array}{c|c} \Sigma_r & 0 \\ \hline 0 & 0 \end{array}\right] ; \Sigma_r = \mathrm{diag}\{\sigma_1, \cdots, \sigma_r\} \tag{9}$$

with the *singular value* σ_i being positive and satisfying $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_r > 0$. The computation of this decomposition is, of course, subject to rounding errors. Denoting computed quantities by an overbar, we generally have for some *error matrix* E_A :

$$\overline{A} = A + E_A = \overline{U\Sigma V}^H \tag{10}$$

The computed decomposition thus corresponds exactly to a *perturbed* matrix $\overline{A}$. When using the SVD algorithm available in the literature [73], this perturbation can be bounded by :

$$\| E_A \| \leq \pi\epsilon \| A \| \tag{11}$$

where ϵ is the machine precision and π some quantity depending on the dimensions m and n, but reasonably close to 1 (see also [105]). Thus, the *backward error* E_A induced by this algorithm, has roughly the same norm as the *input error* E_i that results, for example, when reading the data A into the computer. Then, according to the definition of numerical stability given above, when a bound such as that in (11) exists for the error induced by a numerical algorithm, the algorithm is said to be *backward stable* [198], [41]. Notice that backward stability does not guarantee any bounds on the errors in the result $\overline{U}, \overline{\Sigma}$, and $\overline{V}$. In fact this depends on how perturbations in the data (namely $E_A = \overline{A} - A$) affect the resulting decomposition (namely $E_U = \overline{U} - U, E_\Sigma = \overline{\Sigma} - \Sigma$, and $E_V = \overline{V} - V$). This is commonly measured by the condition $\kappa[f(A)]$ [163].

It is important to note that backward stability is a property of an algorithm while conditioning is associated with a problem and the specific data for that problem. The errors in the result depend on both the stability of the algorithm used and the conditioning of the problem solved. A *good* algorithm should therefore be backward stable since the size of the errors in the result is then mainly due to the condition of the problem, not to the algorithm. An unstable algorithm, on the other hand, may yield a large error even when the problem is well-conditioned.

Bounds of the type (11) are obtained by an error analysis of the algorithm used; see, e.g., [198], [199]. The condition of the problem is obtained by a sensitivity analysis; see, e.g., [198], [193], [173], [178] for some examples.

We close this section with two simple examples to illustrate some of the concepts introduced above.

Example 1: Let x and y be two floating-point computer numbers and let $fl(x * y)$ denote the result of multiplying them in floating-point computer arithmetic. In general, the product $x * y$ will require more precision to be represented exactly than was used to represent x or y. But what can be shown for most computers is that

$$fl(x * y) = x * y(1 + \delta) \tag{12}$$

where $|\delta| < \epsilon$ (= relative machine precision). In other words, $fl(x * y)$ is $x * y$ correct to within a unit in the last place. Now, another way to write (12) is as

$$fl(x * y) = x(1 + \delta)^{1/2} * y(1 + \delta)^{1/2} \tag{13}$$

where $|\delta| < \epsilon$. This can be interpreted as follows: the computed result $fl(x*y)$ is the exact product of the two slightly perturbed numbers $x(1+\delta)^{1/2}$ and $y(1+\delta)^{1/2}$. Note that the slightly perturbed data (not unique) may not even be representable floating-point numbers. The representation (13) is simply a way of accounting for the roundoff incurred in the algorithm by an initial (small) perturbation in the data.

Example 2: Gaussian elimination with no pivoting for solving the linear system of equations

$$Ax = b \tag{14}$$

is known to be numerically unstable. The following data will illustrate this phenomenon. Let

$$A = \begin{bmatrix} 0.0001 & 1.000 \\ 1.000 & -1.000 \end{bmatrix}, b = \begin{bmatrix} 1.000 \\ 0.000 \end{bmatrix}.$$

All computations will be carried out in four-significant-figure decimal arithmetic. The "true answer" $x = A^{-1}b$ is easily seen to be

$$\begin{bmatrix} 0.9999 \\ 0.9999 \end{bmatrix}.$$

Using row 1 as the "pivot row" (i.e., subtracting 10,000 $\times$ row 1 from row 2) we arrive at the equivalent triangular system

$$\begin{bmatrix} 0.0001 & 1.000 \\ 0 & -1.000 \times 10^4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.000 \\ -1.000 \times 10^4 \end{bmatrix}.$$

Note that the coefficient multiplying x_2 in the second equation should be -10,001, but because of roundoff, becomes -10,000. Thus, we compute $x_2 = 1.000$ (a good approximation), but back-substitution in the equation

$$0.0001x_1 = 1.000 - fl(1.000 * 1.000)$$

yields $x_1 = 0.000$. This extremely bad approximation to x_1 is the result of numerical instability. The problem itself can be shown to be quite well-conditioned.

3 Fundamental Problems in Numerical Linear Algebra

In this section we give a brief overview of some of the fundamental problems in numerical linear algebra which serve as building blocks or "tools" for the solution of problems in systems, control, and estimation.

A. Linear Algebraic Equations and Linear Least Squares Problems

Probably the most fundamental problem in numerical computing is the calculation of a vector x which satisfies the linear system

$$Ax = b \tag{15}$$

where $A \in \mathbb{R}^{n\times n}(\text{or}\,\mathbb{C}^{n\times n})$ and has rank n. A great deal is now known about solving (15) in finite arithmetic both for the general case and for a large number of special situations. Some of the standard references include [45], [57], [81], [174], and [73].

The most commonly used algorithm for solving (15) with general A and small n (say $n \leq 200$) is Gaussian elimination with some sort of pivoting strategy, usually "partial pivoting." This essentially amounts to factoring some permutation of the rows of A into the product of a unit lower triangular matrix L and an upper triangular matrix U. The algorithm is effectively stable, i.e., it can be proved that the computed solution is near the exact solution of the system

$$(A + E)x = b \tag{16}$$

with $|e_{ij}| \leq \phi(n)\,\gamma\,\beta\,\epsilon$ where $\phi(n)$ is a modest function of n depending on details of the arithmetic used, γ is a "growth factor" (which is a function of the pivoting strategy and is usually—but not always—small), β behaves essentially like $\|A\|$, and ϵ is the machine precision. In other words,

except for moderately pathological situations, E is "small"—on the order of $\epsilon \|A\|$. See [174], [73] for further details.

The following question then arises. If, because of roundoff errors, we are effectively solving (16) rather than (15), what is the relationship between $(A+E)^{-1}b$ and $A^{-1}b$? To answer this question we need some elementary perturbation theory and this is where the notion of condition number arises. A condition number for the problem (15) is given by

$$\kappa(A) := \|A\| \ \|A^{-1}\|. \tag{17}$$

Simple perturbation results can be used to show that perturbation in A and/or b can be magnified by as much as $\kappa(A)$ in the computed solution. Estimation of $\kappa(A)$ (since, of course, A^{-1} is unknown) is thus a crucial aspect of assessing solutions of (15) and the particular estimation procedure used is usually the principal difference between competing linear equation software packages. One of the more sophisticated and reliable condition estimators presently available is based on [33] and is implemented in LINPACK [45] and its successor LAPACK [3]. In addition to the l_1 condition estimator of [33], LINPACK and LAPACK feature many codes for solving (14) in case A has certain special structures (e.g., banded, symmetric, positive definite).

Another important class of linear algebra problems and one for which codes are available in LINPACK and LAPACK is the linear least squares problem

$$\min \|Ax - b\|_2 \tag{18}$$

where $A \in \mathbb{R}^{m\times n}$ and has rank k, with (in the simplest case) $k = n \leq m$. The solution of (18) can be written formally as $x = A^+ b$. Here, standard references include [45], [123], [73], and [174]. The method of choice is generally based upon the QR factorization of A (for simplicity, let $\text{rank}(A) = n$)

$$A = QR \tag{19}$$

where $R \in \mathbb{R}^{n\times n}$ is upper triangular and $Q \in \mathbb{R}^{m\times n}$ has orthonormal columns, i.e., $Q^T Q = I$. With special care and analysis the case $k < n$ can also be handled similarly. The factorization is effected through a sequence of Householder tranformations H_i applied to A. Each H_i is symmetric and orthogonal and of the form $I - 2uu^T/u^T u$ where $u \in \mathbb{R}^m$ is specially chosen so that zeros are introduced at appropriate places in A when it is premultiplied by H_i. After n such transformations we have

$$H_n H_{n-1} \cdots H_1 A = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

from which the factorization (19) follows. Defining c and d by

$$\begin{bmatrix} c \\ d \end{bmatrix} := H_n H_{n-1} \cdots H_1 b$$

where $c \in \mathbb{R}^n$, it is easily shown that the least squares solution x of (18) is given by the solution of the linear system of equations

$$Rx = c\ . \tag{20}$$

The above algorithm can be shown to be numerically stable and, again, a well-developed perturbation theory exists from which condition numbers can be obtained, this time in terms of

$$\kappa(A) := \|A\| \ \|A^+\|.$$

Least squares perturbation theory is fairly straightforward when $\text{rank}(A) = n$, but is considerably more complicated when A is rank-deficient. The reason for this is that while the inverse is a continuous function of the data (i.e., the inverse is a continuous function in a neighborhood of a nonsingular matrix), the pseudoinverse is discontinuous. For example, consider

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = A^+$$

and perturbations

$$E_1 = \begin{bmatrix} 0 & 0 \\ \delta & 0 \end{bmatrix}$$

and

$$E_2 = \begin{bmatrix} 0 & 0 \\ 0 & \delta \end{bmatrix}$$

with δ small. Then

$$(A + E_1)^+ = \begin{bmatrix} \frac{1}{1+\delta^2} & \frac{\delta}{1+\delta^2} \\ 0 & 0 \end{bmatrix}$$

which is close to A^+ but

$$(A + E_2)^+ = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{\delta} \end{bmatrix}$$

which gets arbitrarily far from A^+ as δ is decreased towards 0. For a complete survey of perturbation theory for the least squares problem and related questions, see [176], [178].

In lieu of Householder transformations, Givens transformations (elementary rotations or reflections) may also be used to solve the linear least squares problem. Details can be found in [45], [123], [157], [175], and [197]. Recently, Givens transformations have received considerable attention for the solution of both linear least squares problems as well as systems of linear equations in a parallel computing environment. The capability of introducing zero elements selectively and the need for only local interprocessor communication make the technique ideal for "parallelization." Indeed, there have been literally dozens of "parallel Givens" algorithms proposed and we include [67], [76], [127], [169], [128], and [141] as representative references.

B. Eigenvalue and Generalized Eigenvalue Problems

In the algebraic eigenvalue/eigenvector problem for $A \in \mathbb{R}^{n\times n}$, one seeks nonzero solutions $x \in \mathbb{C}^n$ and $\lambda \in \mathbb{C}$ which satisfy

$$Ax = \lambda x. \tag{21}$$

The classic reference on the numerical aspects of this problem is Wilkinson [198] with Parlett [157] providing an equally thorough and up-to-date treatment of the case of symmetric A (in which $x \in \mathbb{R}^n, \lambda \in \mathbb{R}$). A more brief textbook introduction is given in [174] [73].

It is really only rather recently that some of the computational issues associated with solving (21) — in the presence of rounding error—have been resolved or even understood. Even now some problems such as the invariant subspace problem continue to be active research areas. For an introduction to some of the difficulties which may be encountered in trying to make numerical sense out of mathematical constructions such as the Jordan canonical form, the reader is urged to consult [74].

The most common algorithm now used to solve (21) for general A is the QR algorithm of Francis [60],[61]. A shifting procedure (see [174], [73] for a brief explanation) is used to enhance convergence

and the usual implementation is called the double-Francis-QR algorithm. Before the QR process is applied, A is initially reduced to upper Hessenberg form $A_H(a_{ij} = 0 \text{ if } i - j \geq 2)$ [130]. This is accomplished by a finite sequence of similarities which can be chosen to be of the Householder form discussed above. The QR process then yields a sequence of matrices which are orthogonally similar to A and which converge (in some sense) to a so-called quasi-upper-triangular matrix S which is also called the real Schur form (RSF) of A. The matrix S is block-upper-triangular with 1×1 diagonal blocks corresponding to real eigenvalues of A and 2×2 diagonal blocks corresponding to complex-conjugate pairs of eigenvalues. The quasi-upper-triangular form permits all arithmetic done to be real rather than complex as would be necessary for convergence to an upper triangular matrix. The orthogonal transformations from both the Hessenberg reduction and the QR process may be accumulated into a single orthogonal transformation U so that

$$U^T A U = R \tag{22}$$

compactly represents the entire algorithm.

An analogous process can be applied in the case of symmetric A and considerable simplifications and specializations result. Moreover, [157], [198], [73], and [178] may be consulted regarding an immense literature concerning stability of the QR and related algorithms, and conditioning of eigenvalues and eigenvectors. Both subjects are vastly more complex for the eigenvalue/eigenvector problem than for the linear equation problem.

Quality mathematical software for eigenvalues and eigenvectors is available; the EISPACK [62], [171] collection of subroutines represents a pivotal point in the history of mathematical software. This collection is primarily based on the algorithms collected in [199]. The successor to EISPACK (and LINPACK) is the recently released LAPACK [3] in which the algorithms and software have been restructured to provide high efficiency on vector processors, high performance workstations, and shared memory multiprocessors.

Closely related to the QR algorithm is the QZ algorithm [143] for the generalized eigenvalue problem

$$Ax = \lambda M x \tag{23}$$

where $A, M \in \mathbb{R}^{n \times n}$. Again, a Hessenberg-like reduction, followed by an iterative process are implemented with orthogonal transformations to reduce (23) to the form

$$QAZy = \lambda QMZy \tag{24}$$

where QAZ is quasi-upper-triangular and QMZ is upper triangular. For a review and references to results on stability, conditioning, and software related to (23) and the QZ algorithm, see [122], [73]. The generalized eigenvalue problem is both theoretically and numerically more difficult to handle than the ordinary eigenvalue problem, but it finds numerous applications in control and system theory [184].

C. The Singular Value Decomposition and Some Applications

One of the basic and most important tools of modern numerical analysis, particularly numerical linear algebra, is the singular value decomposition (SVD). It was introduced in Section 2. Here we make a few comments about its properties and computation as well as its significance in various numerical problems.

Singular values and the singular value decomposition have a long history particularly in statistics and more recently in numerical linear algebra. Even more recently the ideas are finding applications in the control and signal processing literature, although their use there has been overstated

somewhat in certain applications. For a survey of the singular value decomposition, its history, numerical details, and some applications in systems and control theory, see [105].

The fundamental result was stated in Section 2 (for the complex case). The result for the real case is similar and is stated below.

Theorem 1: Let $A \in \mathbb{R}^{m\times n}$ with $\text{rank}(A) = r$. Then there exist orthogonal matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ such that

$$A = U\Sigma V^T \tag{25}$$

where

$$\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}$$

and $\Sigma_r = \text{diag}\,\{\sigma_1, \cdots, \sigma_r\}$ with $\sigma_1 \geq \cdots \geq \sigma_r > 0$.

The proof of Theorem 1 is straightforward and can be found in, for example, [70], [73], and [174]. Geometrically, the theorem says that bases can be found (separately) in the domain and codomain spaces of a linear map with respect to which the matrix representation of the linear map is diagonal. 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 of A^TA. The columns $\{u_k, k = 1. \cdots, m\}$ of U are called the left singular vectors of A (the orthonormal eigenvectors of AA^T), while the columns $\{v_k, k = 1, \cdots, n\}$ of V are called the right singular vectors of A (the orthonormal eigenvectors of A^TA). The matrix A can then also be written (as a dyadic expansion) in terms of the singular vectors as follows:

$$A = \sum_{k=1}^{r} \sigma_k u_k v_k^T.$$

The matrix A^T has m singular values, the positive square roots of the eigenvalues of AA^T. The r [= rank (A)] nonzero singular values of A and A^T are, of course, the same. The choice of A^TA rather than AA^T in the definition of singular values is arbitrary. Only the nonzero singular values are usually of any real interest and their number, given the SVD, is the rank of the matrix. Naturally, the question of how to distinguish nonzero from zero singular values in the presence of rounding error is a nontrivial task.

It is not generally advisable to compute the singular values of A by first finding the eigenvalues of A^TA (remember the folk theorem!), tempting as that is. Consider the following example with μ a real number with $|\mu| < \sqrt{\epsilon}$ (so that $fl(1+\mu^2) = 1$ where $fl(\cdot)$ denotes floating-point computation). Let

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

Then

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

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

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

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. The above example illustrates a potential

pitfall in attempting to form and solve the normal equations in a linear least squares problem, and is at the heart of what makes square root filtering so attractive numerically. Very simplistically speaking, square root filtering involves working directly on an "A-matrix," for example updating it, as opposed to working on (updating, say) an "A^TA-matrix." See [14] for further details and references.

Square root filtering is usually implemented using the QR factorization (or some closely related algorithm) as described previously rather than SVD. The key thing to remember is that in most current computing environments, the condition of the least-squares problem is squared unnecessarily in solving the normal equations. Moreover, critical information may be lost irrecoverably by simply forming A^TA.

Returning now to the SVD there are two features of this matrix factorization that make it so attractive in finite arithmetic: First, it can be computed in a numerically stable way, and second, singular values are well-conditioned. Specifically, there is an efficient and numerically stable algorithm due to Golub and Reinsch [72] (based on [70]) which works directly on A to give the SVD. This algorithm has two phases: In the first phase, it computes orthogonal matrices U_1 and V_1 such that $B = {U_1}^T A V_1$ is in bidiagonal form, i.e., only the elements on its diagonal and first super-diagonal are non-zero. In the second phase, the algorithm uses an iterative procedure to compute orthogonal matrices U_2 and V_2 such that ${U_2}^T B V_2$ is diagonal and non-negative. The SVD defined in (25) is then given by $\Sigma = U^TBV$, where $U = U_1U_2$ and $V = V_1V_2$. 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. Fairly sophisticated implementations of this algorithm can be found in [45] and [62]. The well-conditioned nature of the singular values follows from the fact that if A is perturbed to $A+E$, then it can be proved that

$$|\sigma_i(A+E) - \sigma_i(A)| \leq \|E\|.$$

Thus, the singular values are computed with small absolute error although the relative error of sufficiently small singular values is not guaranteed to be small. A new algorithm for computing the singular values of a bidiagonal matrix [44] overcomes this deficiency to the extent that it computes the singular values of a bidiagonal matrix to the same relative precision as that of the individual matrix entries. In other words, the algorithm will obtain accurate singular values from accurate bidiagonal matrices. However, one cannot in general guarantee high accuracy in the reduction to bidiagonal form.

It is now acknowledged that the singular value decomposition is the most generally reliable method of determining rank numerically (see [74] for a more elaborate discussion). However, it is considerably more expensive to compute than, for example, the QR factorization which, with column pivoting [45], can usually give equivalent information with less computation. Thus, while the SVD is a useful theoretical tool, its use for actual computations should be weighed carefully against other approaches.

Only rather recently has the problem of numerical determination of rank become well-understood. A recent treatment of the subject can be found in the paper by Chan [26]; see also [177]. The essential idea is to try to determine a "gap" between "zero" and the "smallest nonzero singular value" of a matrix A. Since the computed values are exact for a matrix near A, it makes sense to consider the rank of all matrices in some δ-ball (with respect to 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. We refer to [177] for further details. We must emphasize, however, that even with SVD, numerical determination of rank in finite arithmetic is a highly nontrivial problem.

That other methods of rank determination are potentially unreliable is demonstrated by the following example which is a special case of a general class of matrices studied by Ostrowski [150]. 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 $(= -1)$ are well away from zero. Its determinant is $(-1)^n$—definitely not close to zero. But this matrix is, in fact, very nearly singular and gets more nearly so as n increases. Note, for example, that

$$\begin{bmatrix} -1 & +1 & \cdots & & \cdots & +1 \\ 0 & \ddots & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & +1 \\ 0 & \cdots & & \cdots & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 2^{-1} \\ \vdots \\ 2^{-n+1} \end{bmatrix}$$

$$= \begin{bmatrix} -2^{-n+1} \\ -2^{-n+1} \\ \vdots \\ -2^{-n+1} \end{bmatrix} \to \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} (n \to +\infty).$$

Moreover, adding 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 $\sigma_n(A)$ behaves as 2^{-n+1}. A corollary for control theory: eigenvalues do not necessarily give a reliable measure of "stability margin." As an aside it is useful to note here that in this example of an invertible matrix, the crucial quantity, $\sigma_n(A)$, which measures nearness to singularity, is simply $1/\|A^{-1}\|$, and the result is familiar from standard operator theory. There is nothing intrinsic about singular values in this example and, in fact, $\|A^{-1}\|$ might be more cheaply computed or estimated in other matrix norms. This is precisely what is done in estimating the condition of linear systems in LINPACK where $\|\cdot\|_1$ is used [33].

Since rank determination, in the presence of roundoff error, is a nontrivial problem, all the same difficulties will naturally arise in any problem equivalent to or involving rank determination, such as determining the independence of vectors, finding the dimensions of certain subspaces, etc. Such problems arise as basic calculations throughout systems, control, and estimation theory. Selected applications are discussed in more detail in [105].

Finally, let us close this section with a brief example illustrating a totally inappropriate use of SVD. The rank condition

$$\text{rank } [B, AB, \cdots, A^{n-1}B] = n \tag{26}$$

for the controllability of (1) is too well-known. Suppose

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

with $|\mu| < \sqrt{\epsilon}$. Then

$$fl[B, AB] = \begin{bmatrix} 1 & 1 \\ \mu & \mu \end{bmatrix}$$

and now even applying SVD, the erroneous conclusion of uncontrollability is reached. Again the problem is in just forming AB; not even SVD can come to the rescue after that numerical *faux pas*.

4 Applications to Systems and Control

A reasonable approach to developing numerically reliable algorithms for computational problems in linear system theory would be to reformulate the problems as concatenations of subproblems for which numerically stable algorithms are available. Unfortunately, one cannot ensure that the stability of algorithms for the subproblems results in stability of the overall algorithm. This requires separate analysis which may rely on the sensitivity or conditioning of the subproblems. In the next section we show that delicate (i.e., badly conditioned) subproblems should be avoided whenever possible; a few examples are given where a possibly badly conditioned step is circumvented by carefully modifying or completing existing algorithms (see also [183], [184]).

A second type of difficulty is the ill-posedness of some of the problems occurring in linear system theory [201]. Two approaches can be adopted here. One can develop an acceptable perturbation theory for such problems, making use of a concept such as *restricted conditioning* which is conditioning under perturbations for which a certain property holds, e.g., fixed rank [89], [184]. One then looks for restricting assumptions that make the problem well-posed. Another approach is to delay any such *restricting choices* to the end and leave it to the user as to what choice to make by looking at the results. The algorithm then provides quantitative measures that help the user make this choice (see, e.g., [148], [111], [147]). By this approach one may avoid artificial restrictions of the first approach that sometimes do not respect the practical significance of the problem.

A third possible *pitfall* is that many users almost always prefer fast algorithms to slower ones. However, it is often the case that slower algorithms are more reliable; see, e.g., [42], [21].

In the subsections that follow, we survey a representative selection of numerical linear algebra problems arising in linear systems, control, and estimation theory, which have been examined using some of the techniques described in the preceding sections. Many of these topics are discussed in more detail in the papers included in this book. It is worth noting that some of the scalar algorithms that are discussed here do not extend trivially to the matrix case. When they do, we shall only mention the matrix case. We shall only discuss the numerical aspects here; for the system theoretical background, we refer the reader to the control and systems literature.

4.1 Some Typical Techniques

Most of the reliable techniques in numerical linear algebra are based on the use of orthogonal transformations. Typical examples of this are the QR decomposition for least squares problems, the Schur decomposition for eigenvalue and generalized eigenvalue problems, and the singular value decomposition for rank determinations and generalized inverses. The use of orthogonal transformations is also very much in evidence in most of the reliable linear algebra techniques for control theory. This is partly due to the direct application of existing linear algebra decompositions to problems in control. Obvious examples of this are the Schur approach for solving algebraic Riccati equations, both continuous and discrete time [111], [156], [185], for solving Lyapunov equations [12], and for performing pole placement [194]. But new orthogonal decompositions have also been introduced which rely heavily on the same principles, but were specifically developed for problems encountered in control. Orthogonal state-space transformations on a system $\{A, B, C\}$ result in a new state-space representation $\{U^H AU, U^H B, CU\}$ where U is chosen to perform some kind of decomposition on the matrices A, B, and C. These special forms have been termed "condensed forms" and include:

- the state Schur form [194],

- the state Hessenberg form [115],
- the observer Hessenberg form [187], [139],
- the controller Hessenberg form [191], [134], [159].

Staircase forms or block Hessenberg forms are other variants of these condensed forms which have proven to be very useful in dealing with MIMO systems [184], [158], [151], [106].

The main reasons for using these orthogonal state-space transformations are twofold:

- The numerical sensitivity of the control problem being solved is not affected by these transformations. This is because sensitivity is typically measured by norms or angles of certain spaces and such things are unaltered by orthogonal transformations.
- Orthogonal transformations have minimum condition number, which is essential in proving bounded error propagation, and establishing numerical stability of the algorithm that uses such transformations.

More details on this are given in the paper [190], and in subsequent sections where some of these condensed forms are used for particular applications.

4.2 Transfer Functions, Poles and Zeros

In this section, we discuss important structural properties of linear systems and the numerical techniques that are available for determining them. The transfer function $R(\lambda)$ of a linear system is given by a polynomial representation $V(\lambda)T^{-1}(\lambda)U(\lambda) + W(\lambda)$ or by a state-space model $C(\lambda I - A)^{-1}B + D$. The results in this subsection hold for both the discrete-time case (where λ stands for the shift operator z) and the continuous-time case (where λ stands for the differentiation operator D).

A. The Polynomial Approach (see [107] and references therein)

One is interested in a number of structural properties of the transfer function $R(\lambda)$ such as poles, transmission zeros, decoupling zeros, etc. In the scalar case, where $\{T(\lambda), U(\lambda), V(\lambda), W(\lambda)\}$ are scalar polynomials, all this can be found with a GCD extraction routine and a rootfinder, for which reliable methods exist; see, e.g., [107], [83], [16], [78], and references therein. In the matrix case the problem becomes much more complex and the basic method for GCD extraction — the Euclidean algorithm — becomes unstable (see [184]). Moreover, other structural elements (null spaces, etc.) come into the picture, which makes the polynomial approach less attractive than the state-space approach [184], [189].

B. The State-Space Approach (see [201], [184], and references therein)

The structural properties of interest are poles and zeros of $R(\lambda)$, decoupling zeros, controllable and unobservable subspaces, supremal (A, B)-invariant and controllability subspaces, factorizability of $R(\lambda)$, left and right null spaces of $R(\lambda)$, etc. These concepts play a fundamental role in several design problems and have received considerable attention over the last few years [170], [184], [151], [105], [148], [111], [147], [172], [189]. In [184] and [186], it is shown that all the concepts mentioned above can be considered as generalized eigenstructure problems and that they can be computed via the Kronecker canonical form of the pencils

$$\begin{bmatrix} \lambda I - A \end{bmatrix} \qquad \begin{bmatrix} \lambda I - A \mid B \end{bmatrix}$$
$$\left[\begin{array}{c} \lambda I - A \\ \hline -C \end{array}\right] \qquad \left[\begin{array}{c|c} \lambda I - A & B \\ \hline -C & D \end{array}\right] \tag{27}$$

or from other pencils derived from them. Backward stable software is also available for computing the Kronecker structure of an arbitrary pencil [183], [13]. A remaining problem here is that the determination of several of the structural properties listed above may be ill-posed in some cases and that one has to develop the notion of restricted conditioning in these cases (see [184]). Sensitivity results in this area are still few but are slowly emerging [183], [43]. A completely different approach is to reformulate the problem as an approximation or optimization problem for which *quantitative measures* are derived, leaving the final choice to the user. Results in this vein are obtained for controllability, observability [147], [151], [158], (almost) (A, B)-invariant and controllability subspaces [148], [43].

4.3 Controllability and Other "Abilities"

Basic to the study of linear control and system theory are the various "abilities" such as controllability, observability, reachability, reconstructibility, stabilizability, and detectability [201]. These concepts can also be viewed in terms of the concepts of decoupling zeros, controllable and unobservable subspaces, controllability subspaces, etc. that were mentioned in the previous section. Our remarks here are confined, but not limited, to the notion of controllability.

A large number of algebraic and dynamic characterizations of controllability have been given; see [114] for a sample. But each and every one of these has difficulties when implemented in finite arithmetic. For a survey of this topic and numerous examples, see [151]. Part of the difficulty in dealing with controllability numerically lies in the intimate relationship with the invariant subspace problem [74]. The controllable subspace associated with (1) is the smallest A-invariant subspace (subspace spanned by eigenvectors or principal vectors) containing the range of B. Since A-invariant subspaces can be extremely sensitive to perturbation, it follows that, so too, is the controllable subspace. Similar remarks apply to the computation of the so-called controllability indices. The example discussed in the third paragraph of Section 2 can be used to illustrate these remarks dramatically. The matrix A has but one eigenvector (associated with 0) whereas the slightly perturbed A has n eigenvectors associated with the n distinct eigenvalues.

Recently, attempts have been made to provide numerically stable algorithms for the pole placement problem. This problem will be discussed in a later section. It suffices to mention here that the problem of pole placement by state feedback is closely related to controllability. Recent work on developing numerically stable algorithms for pole placement is based on the reduction of A to a Hessenberg form; see, e.g., [134], [159]. In the single-input case, a good approach is to use the controller Hessenberg form mentioned in Section 4.1, where the state matrix A is upper Hessenberg and the input vector B is a multiple of $(1, 0, \cdots, 0)^T$. The pair (A, B) is then controllable if and only if all $(n-1)$ subdiagonal elements of A are nonzero. If a subdiagonal element is 0, the system is uncontrollable and a basis for the uncontrollable subspace is easily constructed. The transfer function gain or first nonzero Markov parameter is also easily constructed from this "canonical form." In fact, the numerically more robust system Hessenberg form is playing and will continue to play an ever-increasing role in system theory in replacing the numerically more fragile special case of the companion or rational canonical or Luenberger canonical form.

A more important aspect of controllability is to better understand topological notions such as "near-uncontrollability." But there are numerical difficulties lurking here also, and we refer to Parts 3 and 4 for further details. Related to this is an interesting system-theoretic concept called "balancing" which is discussed in the paper by Moore in Part 3. The computation of "balancing transformations" is discussed in [112], [121].

There are at least two distinct notions of near-uncontrollability [114]: in the parametric sense and in the energy sense. In the parametric sense a controllable pair (A, B) is said to be near-

uncontrollable if the parameters of (A, B) need be perturbed by only a relatively small amount for (A, B) to become uncontrollable. In the energy sense, a controllable pair is near-uncontrollable if large amounts of control energy ($\int u^T u$) are required to effect a state transfer. The pair

$$A = \begin{pmatrix} 0 & 1 & & \cdots & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ & & & & \\ \vdots & & & \ddots & 1 \\ 0 & \cdots & & \cdots & 0 \end{pmatrix}, \; B = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

is very near uncontrollable in the energy sense but not as badly so in the parametric sense. Of course, both measures are coordinate dependent and "balancing" is one attempt to try to remove this coordinate bias. The pair (A, B) above is in "controllable canonical form", and it is now known that matrices in this form (specifically, the A matrix in rational canonical form) almost always exhibit poor numerical behavior and are "close to" uncontrollable (unstable, etc.) as the size n increases. For details, see [94].

4.4 Identification

A. Realization ([42] and references therein).

Let *R(z)* be an $m \times n$ transfer function of a discrete-time causal system, and let its impulse response be given by

$$R(z) \; = \; \sum_{i=0}^{\infty} H_i \, z^{-i}. \tag{28}$$

The realization problem is to find the transfer function $R(z)$ in, e.g., polynomial description $P(z)Q^{-1}(z)$ or state-space description $R(z) = D + C(zI - A)^{-1}B$, when the impulse response sequence $\{H_i\}$ is given. In a later stage one might also be interested in the poles and zeros of $R(z)$. In the scalar case this problem is in fact the Padé approximation problem, for which fast methods exist (see [42], [20] for a survey). In [42], it is shown that these methods are unstable and that they all reduce to factoring the Hankel matrix

$$H = \begin{bmatrix} H_1 & H_2 & \cdots & H_n \\ H_2 & \iddots & \iddots & \vdots \\ \vdots & \iddots & \iddots & \vdots \\ H_n & \cdots & \cdots & H_{2n-1} \end{bmatrix} \tag{29}$$

where n is an upper bound for the McMillan degree of $R(z)$. In [42] a stable but slower algorithm is given for finding a polynomial and a state-space description of $R(z)$. A typical state-space method based on the singular value decomposition of H was given in [204] and shown to be stable in [108]. As shown in [147], [108] the realization $\{\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D}\}$ computed is *balanced*, which has the nice property of being less sensitive to rounding errors in H_i than the models used in [42]. Here a typical example can be given of the first difficulty discussed in the introduction to Section 4. For computing the poles of $R(z)$ one could compute its Padé approximation and then compute the zeros of $Q(z)$. The construction of this intermediate result can be very sensitive to rounding errors in H_i, which endangers the computation of the zeros of $Q(z)$. As reported in [147], [108], the method using the eigenvalues of A is less prone to rounding errors. This problem is also related to that of

optimal approximation in the Hankel norm [1], [68], and has led to a renewed interest in the partial realization problem and related topics [4].

Very often one does not have access to the impulse response of the system but only to a sequence of inputs $\{u_i\}$ and corresponding outputs $\{y_i\}$. In such cases a novel algorithm was recently derived [145] based on the following Hankel matrix:

$$H = \begin{bmatrix} z_1 & z_2 & \cdots & z_m \\ z_2 & \iddots & \iddots & \vdots \\ \vdots & \iddots & \iddots & \vdots \\ z_k & \cdots & \cdots & z_{m+k-1} \end{bmatrix}, \quad \text{where } z_i = \begin{bmatrix} u_i \\ y_i \end{bmatrix}. \tag{30}$$

The matrix is split rowwise into two blocks H_{up} and H_{lo} and the intersection of the rowspaces of these two blocks is then computed which amounts to a particular rank factorization. In the case that the input-output pairs are sufficiently "rich" signals (called *persistence of excitation*) one can show [145] that the dimension of this intersection is the state dimension of the underlying state-space model, and one can then identify the matrices $\{A, B, C, D\}$ via a least squares problem derived from these factorizations.

Both the above approaches are very general in the sense that they apply to MIMO systems. In the case of SISO systems several methods have been derived that also assume a particular noise model and then reduce the identification problem to a minimum variance problem of the noise process (see [126] for a survey). Only in a special case does this actually reduce to a least-squares problem that can be solved in much the same way as the above Hankel I/O method. In general, though, the minimization of the noise process reduces to a nonlinear minimization which can only be solved iteratively. Moreover, extensions of these approaches to the MIMO case considerably increase the complexity of this minimization approach.

B. Linear prediction ([129] and references therein)

The problem here is to model a given time signal $\{y_k\}$ with z transform $Y(z)$ as the output of a linear time-invariant system

$$Y(z) = \frac{g}{a(z)} U(z) \tag{31}$$

where $a(z) = 1 + a_1 z^{-1} + \cdots + a_p z^{-p}$ and $U(z)$ is the z transform of a periodic pulse train or of white noise, depending on the application [129]. The so-called autocorrelation method converts this problem to one of factoring or inverting the positive definite Toeplitz matrix

$$T = \begin{bmatrix} r(0) & \cdots & r(p) \\ \vdots & \ddots & \vdots \\ r(p) & \cdots & r(0) \end{bmatrix}; \; r(i) = \sum_{n=-\infty}^{+\infty} y_n y_{n+i} \tag{32}$$

where $r(i)$ is the autocorrelation function of the signal $\{y_k\}$. The Cholesky factorization of this matrix requires $O(n^3)$ operations, while the Levinson-Durbin algorithm requires only $O(n^2)$ operations by efficiently exploiting the Toeplitz structure of T. For quite some time the stability and conditioning issues of this approach have been a point of controversy [37]. However, a rigorous numerical treatment of the problem has shown that the algorithm is reasonably stable, but that the problem is quite often badly conditioned [37], [19]. We notice that the problem (31) is in fact an approximation problem, whose solution is given by the above Toeplitz factorization. An error

analysis in [37], [19] only refers to errors in the factorization, not to the approximation error. We point out that the *lattice algorithm* of Itakura and Saito [82] — which computes the same Cholesky factor by directly working on the data $\{y_i\}$ arranged in a Toeplitz matrix — is also stable in the same sense [38]. Moreover, it avoids the construction of the Toeplitz matrix T which could lead to problems in extremely badly conditioned cases (see [38] for more details).

C. Spectral factorization (see [200], [2] and references therein).

An alternative to (31) is to allow for a more general type of transfer function between $Y(z)$ and $U(z)$:

$$Y(z) = g\frac{b(z)}{a(z)}U(z). \tag{33}$$

This can now be converted to a spectral factorization problem [200], [2] of the rational transfer function

$$\Phi(z) = R_p(z) + R_p(z^{-1}) = g^2\,\frac{b(z^{-1})b(z)}{a(z^{-1})a(z)} \tag{34}$$

where $R_p(z)$ is a *p-th* order approximation of the impulse response $R(z)$:

$$R(z) \approx \frac{r(0)}{2} + r(1)\,z^{-1} + r(2)\,z^{-2} + \cdots \tag{35}$$

Here again the original problem is one of approximation for which (34), (35) is a proposed solution. A polynomial approach would yield $a(z)$ during the realization step (35), and $b(z)$ during the spectral factorization step, which can be converted to a polynomial spectral factorization problem (see, e.g., [107]):

$$\Phi_N(z) = g^2\,b(z^{-1})\,b(z); \tag{36}$$

where

$$\Phi_N(z) = \Phi_p z^{-p} + \cdots + \Phi_1 z^{-1} + \Phi_0 + \Phi_1 z + \cdots + \Phi_p z^p \tag{37}$$

is derived from $\Phi(z)$.

Bauer has shown that $b(z)$ can be obtained from the rows of the Cholesky factor B of the infinite (positive definite) Toeplitz matrix

$$T_\Phi = \begin{bmatrix} \Phi_0 & \Phi_1 & \cdots & \Phi_p & 0 \\ \Phi_1 & \Phi_0 & \ddots & & \ddots \\ \vdots & \ddots & \ddots & & \\ \Phi_p & & & & \\ & \ddots & & & \\ 0 & & & & \end{bmatrix} = B^T B\,;\; B = \begin{bmatrix} b_0^{(1)} & b_1^{(1)} & \cdots & b_p^{(1)} & & \\ & b_0^{(2)} & b_1^{(2)} & \cdots & b_p^{(2)} & 0 \\ & & \ddots & \ddots & & \ddots \\ & 0 & & & & \end{bmatrix}. \tag{38}$$

Unfortunately, the convergence of $b^{(i)}(z)$ to $b(z)$ and the conditioning of the solution $b(z)$ is rather bad when some of the roots of $b(z)$ get close to the unit circle. The same problems occur with the improved Bauer algorithm [166], and with Vostry's Newton-type scheme [196]:

$$b^{(i+1)}(z) = \frac{1}{2}[b^{(i)}(z) + x^{(i)}(z)] \tag{39}$$

where

$$b^{(i)}(z^{-1})x^{(i)} + b^{(i)}(z)x^{(i)}(z^{-1}) = 2\Phi_N(z). \tag{40}$$

The solution of (39), (40) requires $O(p^2)$ operations per iteration step and the convergence of $b^{(i)}(z)$ to $b(z)$ is quadratic when no roots of $b(z)$ are close to the unit circle. The stability of Vostry's method is good since it is a method of iterative refinement using the original data $\Phi_N(z)$ for computing the correction [198]. A third method — suffering from similar problems of convergence — is the one described by Henrici using the FFT algorithm [78], but it also allows for an implicit stability criterion.

A completely different and conceptually very simple approach is to compute the roots of $z^p\Phi_N(z)$ and perform the factorization by grouping the roots inside the unit circle. This method is quite fast — $O(n^2)$ operations independently of the location of the zeros — and backward stable when the deflations are implemented carefully [83]. A similar approach in the state-space domain has been developed recently [186], and is based on the computation of a stable deflating subspace of the pencil

$$z\begin{bmatrix} 0 & 0 & 0 \\ 0 & A^T & 0 \\ 0 & B^T & 0 \end{bmatrix} - \begin{bmatrix} A & 0 & B \\ 0 & I & C^T \\ C & 0 & D + D^T \end{bmatrix} \tag{41}$$

where $R_p(z) = C(zI_p - A)^{-1}B + D$. Backward stable software for solving this generalized eigenvalue problem can also be found in [186]. The method also applies to the matrix case and avoids the construction of $\Phi_N(z)$. In the matrix case, this approach is to be preferred over the matrix generalization of (38) and (39), (40) — such as that described in [84] — since, in the latter there is no guarantee of numerical stability.

4.5 Computation of Objects Arising in the Geometric Theory of Linear Multivariable Control

A great many numerical problems arise in the geometric approach [201] to control of systems modeled as (1), (2). Some of these are discussed in the papers by Klema and Laub [105] and Van Dooren [184] in Part 2. Wonham's text [201] remains a fertile source of numerical problems. In fact, the power of the geometric approach derives in large part from the fact that it is independent of specific coordinate systems or matrix representations. Numerical issues are a separate concern. Two of the more elaborate but still fundamental objects in that theory are supremal (A, B)-invariant and controllability subspaces contained in a given subspace. An initial attempt at characterizing these spaces in terms of eigenvectors and a generalized eigenvalue problem was given in [148].

However, a rather different and very thorough numerical treatment of the problem has been given by Van Dooren [183], [184], [88]. This work has implications for most calculations done with linear state-space models. For example, one byproduct is an extremely reliable algorithm (which is similar to an orthogonal version of Silverman's structure algorithm) for the computation of multivariable system zeros [51]. Like [122] this method involves a generalized eigenvalue problem (the Rosenbrock pencil) but the "infinite zeros" are first removed by deflating the given matrix pencil.

4.6 Frequency Response Calculations

Many properties of a linear system (1), (2) are known in terms of its frequency response matrix

$$G(j\omega) := C(j\omega I - A)^{-1}B + D; \qquad (\omega \geq 0) \tag{42}$$

(or $G(e^{j\theta})$; $\theta \in [0, 2\pi]$ for (3), (4)). In fact, various norms of the return difference matrix $I + G(j\omega)$ and related quantities have been investigated in control and system theory as providing measures of robustness of a linear system with respect to stability, noise response, disturbance attenuation, sensitivity, etc. See [113] for some of the numerical aspects and [50], [168] for surveys of some of the control aspects.

It is thus a problem of considerable computational interest to compute $G(j\omega)$ efficiently, given A, B, and C for a (possibly) large number of values of ω (for convenience we shall take D to be 0 since if it is nonzero it is trivial to add to G). An efficient and generally applicable algorithm for this problem is presented in [115]. Rather than solving the linear equation $(j\omega I - A)X = B$ with dense unstructured A, which would require $O(n^3)$ operations for each successive value of ω, the new method does an initial reduction of A to upper Hessenberg form H. The orthogonal state-space coordinate transformations used to obtain the Hessenberg form of A are incorporated into B and C giving $\tilde{B}$ and $\tilde{C}$. Now as ω varies, the coefficient matrix in the linear equation $(j\omega I - H)X = \tilde{B}$ remains in upper Hessenberg form. The advantage is that X can now be found in $O(n^2)$ operations rather than $O(n^3)$ as before, a substantial saving. Moreover, the method is numerically very stable (via either LU or QR factorization) and has the advantage of being independent of the eigenstructure (possibly ill-conditioned) of A. Portable mathematical software, in a sense to be discussed later, is also available for this problem [118]. A systolic version of the algorithm based on QR factorization is described in [25]. Another efficient and reliable algorithm for frequency response computation [139] uses the observer Hessenberg form mentioned in Section 4.1 together with a determinant identity and a property of the LU decomposition of a Hessenberg matrix.

We note here that the above methods can also be extended to state-space models in implicit form, i.e., where (1) is replaced by

$$E\dot{x} = Ax + Bu. \tag{43}$$

Then (42) is replaced with

$$G(j\omega) = C(j\omega E - A)^{-1}B + D \tag{44}$$

and the initial triangular/Hessenberg reduction employed in [143], or the condensed form in [138], can be employed to again reduce the problem to one of updating the diagonal of a Hessenberg matrix and consequently an $O(n^2)$ problem.

A recent advance for the frequency response evaluation problem is the use of matrix interpolation methods to achieve even greater computational efficiency. The basic algorithm is described in [101] while its extension to the implicit or descriptor form is in [180].

4.7 Numerical Solution of Linear Ordinary Differential Equations and Matrix Exponentials

The "simulation" or numerical solution of linear systems of ordinary differential equations (ODE's) of the form

$$\dot{x}(t) = Ax(t) + f(t); \qquad x(0) = x_0 \tag{45}$$

is a standard problem that arises in finding the time response of a system given in state-space form. However, there is still debate as to what is the most effective numerical algorithm, particularly when A is defective (a deficiency of eigenvectors) or nearly defective. The most common approach involves computation of the matrix exponential e^{tA}, since the solution of (45) can be written simply as

$$x(t) = e^{tA}x_0 + \int_0^t e^{(t-s)A}f(s)\,ds.$$

A delightful survey of computational techniques for matrix exponentials is given in [144]. Nineteen "dubious" ways are explored (there exist many more ways which are not discussed) but no clearly superior algorithm is singled out. Methods based on Padé approximation or reduction of A to real Schur form are seen as generally attractive while methods based on Taylor series or the characteristic polynomial of A are generally found to be unattractive. An interesting open problem is the design of a special algorithm for the matrix exponential when the matrix is known *a priori* to be stable ($\Lambda(A)$ in the left half of the complex plane).

The reason for the adjective "dubious" in the title of [144] is that in many (maybe even most) circumstances, it is better to treat (45) as a system of differential equations, typically stiff, and to apply various ODE techniques, specially tailored to the linear case. This approach is discussed in [52], [30], [195]. ODE techniques are clearly to be preferred when A is large and sparse for, in general, e^{tA} will be unmanageably large and dense. The relationship between ODE techniques and matrix exponential techniques when A has an ill-conditioned eigenstructure or when the "exponential problem" is ill-conditioned [87], [193] is not well understood.

4.8 Lyapunov, Sylvester, and Riccati Equations

Certain matrix equations arise naturally in linear control and system theory. Among those frequently encountered in the analysis and design of continuous-time systems are the Lyapunov equation

$$AX + XA^T + Q = 0, \tag{46}$$

and the Sylvester equation

$$AX + XF + Q = 0. \tag{47}$$

The appropriate discrete-time analogs are

$$AXA^T - X + Q = 0 \tag{48}$$

and

$$AXF - X + Q = 0. \tag{49}$$

Various hypotheses are made on the coefficient matrices A, F, Q to ensure certain properties of the solution X.

There is a voluminous literature in control and system theory on these equations but most of that is *ad hoc*, at best, from a numerical point of view, with little attention paid to questions of numerical stability, conditioning, machine implementation, and the like.

For the Lyapunov equation the overall best algorithm in terms of efficiency, accuracy, reliability, availability, and ease of use appears to be that of Bartels and Stewart [12]. The basic idea is to reduce A to quasi-upper-triangular form [or real Schur form (RSF)] and perform a back substitution for the elements of X.

An attractive algorithm for solving Lyapunov equations has been proposed by Hammarling [75]. This algorithm is a variant of the Bartels-Stewart algorithm but instead solves directly for the Cholesky factor Y of X: $Y^TY = X$ and Y is upper triangular. Clearly, given Y, X is easily recovered if necessary. But in many applications, for example [121], only the Cholesky factor is required.

For the Lyapunov equation, when A is stable, the solutions of the above equations are also equal to the reachability and observability Grammians $P_r(T)$ and $P_o(T)$, respectively, for $T = +\infty$ for the system $\{A, B, C\}$:

$$\begin{array}{rclcrcl} P_r(T) & = & \int_0^T e^{tA}BB^T e^{tA^T}\,dt & ; & P_o(T) & = & \int_0^T e^{tA^T}C^TCe^{tA}dt \\ P_r(T) & = & \sum_{k=0}^T A^kBB^T(A^T)^k & ; & P_o(T) & = & \sum_{k=0}^T (A^T)^kC^TCA^k. \end{array} \tag{50}$$

These can be used along with some additional transformations (see [147], [108], [112], [121]) to compute so-called *balanced* realizations $\{\tilde{A}, \tilde{B}, \tilde{C}\}$. For these realizations both P_o and P_r are equal and diagonal. These realizations have some nice sensitivity properties with respect to poles, zeros, truncation errors in digital filter implementations, etc. [147], [108], [149]. They are therefore recommended whenever the choice of a realization is left to the user. When A is not stable one can still use the *finite range* Grammians (50) for $T < +\infty$ for the purpose of balancing [147]. A reliable method for computing integrals and sums of the type (50) can be found in [192], [5]. A sensitivity analysis of the matrix exponential can be found in [193], [95], [99]. It is also shown in [147] that the reachable subspace and the unobservable subspace are the image and the kernel of $P_r(T)$ and $P_o(T)$, respectively. From these relationships, sensitivity properties of the spaces under perturbations of $P_r(T)$ and $P_o(T)$ can be derived.

For the Sylvester equation, the Bartels-Stewart algorithm reduces both A and F to real Schur form (RSF) and then a back substitution is done. It has been demonstrated in [71] that some improvement in this procedure is possible by only reducing the larger of A and F to upper Hessenberg form. The stability of this method has been analyzed in [71]. Although only *weak stability* is obtained, this is satisfactory in most cases.

Algorithms and software for the more general Sylvester equation

$$A_1XF_1{}^T + A_2XF_2{}^T + Q = 0$$

and its symmetric Lyapunov counterpart

$$AXF^T + FXA^T + Q = 0$$

are given in [65], [66].

Open questions remain concerning estimating the condition of Lyapunov and Sylvester equations efficiently and reliably in terms of the coefficient matrices, but one major result has been obtained in [80]. There the condition of, say, (46) is related to the norm of the solution of the same equation with Q replaced by I. Many other interpretations of this fundamental result are also given.

A deeper analysis of the Lyapunov and Sylvester equations is probably a prerequisite to at least a better understanding of conditioning of the Riccati equation for which again, there is considerable theoretical literature but not as much known from a purely numerical point of view. The symmetric $n \times n$ algebraic Riccati equation takes the form

$$Q + AX + XA^T - XGX = 0 \tag{51}$$

for continuous-time systems and

$$A^TXA - X - A^TXG_1(G_2 + G_1^TXG_1)^{-1}G_1^TXA + Q = 0 \tag{52}$$

for discrete-time systems. These equations appear in several design/analysis problems such as optimal control, optimal filtering, spectral factorization; see, e.g., [11], [187], [185], [111], [156], [23] and references therein. Again, appropriate assumptions are made on the coefficient matrices to guarantee the existence and/or uniqueness of certain kinds of solutions X. Nonsymmetric Riccati equations of the form

$$Q + A_1X + XA_2 - XGX = 0 \tag{53}$$

for the continuous-time case (along with an analog for the discrete-time case) are also studied and can be solved numerically by the techniques discussed below.

Several algorithms have been proposed based on different approaches [11], [185], [111], [23]. One of the more reliable general-purpose methods for solving Riccati equations is the Schur method [110], [111]. For the case of (51), for example, this method is based upon the reduction of the associated $2n \times 2n$ Hamiltonian matrix

$$\begin{pmatrix} A & -G \\ -Q & -A^T \end{pmatrix} \tag{54}$$

to RSF. If the RSF is ordered so that its stable eigenvalues (there will be exactly n of them under certain standard assumptions) are in the upper left corner, the corresponding first n vectors of the orthogonal matrix which effects the reduction will form a basis for the stable eigenspace from which the nonnegative definite solution X is then easily found.

Extensions to the basic Schur method have been made [156], [185] which were motivated by the following situations:

- G in (51) is of the form $BR^{-1}B^T$ where R may be nearly singular, or G_2 in (52) may be exactly or nearly singular.
- A in (52) is singular [A^{-1} is required in the classical approach involving a symplectic matrix which plays a role analogous to (54)].

This resulted in the generalized eigenvalue approach requiring the computation of a basis for the deflating subspace corresponding to the stable generalized eigenvalues. For the solution of (51), the generalized eigenvalue problem is given by

$$\lambda \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} A & 0 & B \\ -Q & -A^T & 0 \\ 0 & B^T & R \end{bmatrix}; \tag{55}$$

while for (52), the corresponding problem is

$$\lambda \begin{bmatrix} I & 0 & 0 \\ 0 & A^T & 0 \\ 0 & {G_1}^T & 0 \end{bmatrix} - \begin{bmatrix} A & 0 & -G_1 \\ -Q & I & 0 \\ 0 & 0 & G_2 \end{bmatrix}. \tag{56}$$

The above extensions can be generalized even further [117], as the following problem will illustrate. Consider the optimal control problem

$$\min \frac{1}{2} \int_0^{+\infty} [x^T Q x + 2x^T S u + u^T R u] dt \tag{57}$$

$$\text{subject to } E\dot{x} = Ax + Bu. \tag{58}$$

The Riccati equation associated with (57), (58) then takes the form

$$E^T X B R^{-1} B^T X E - (A - BR^{-1}S^T)^T X E - E^T X (A - BR^{-1}S^T) - Q + SR^{-1}S^T = 0 \tag{59}$$

or

$$(E^T X B + S) R^{-1} (B^T X E + S^T) - A^T X E - E^T X A - Q = 0. \tag{60}$$

This so-called "generalized" Riccati equation can be solved by considering the associated matrix pencil

$$\begin{pmatrix} A & 0 & B \\ -Q & -A^T & -S \\ S^T & B^T & R \end{pmatrix} - \lambda \begin{pmatrix} E & 0 & 0 \\ 0 & E^T & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{61}$$

Note that S in (57) and E in (58) are handled directly and no inverses appear. It is worth emphasizing here that the presence of a nonsingular E in state-space models of the form (58) adds no particular difficulty to the solution process and is numerically the preferred form if E is, for example, near-singular or even sparse. Similar remarks apply to the frequency response problem in (43), (44) and, indeed, throughout all of linear control and system theory. The stability and conditioning of these approaches are discussed in [185], [111], [174] while scaling strategies are addressed in [102]. Other methods, including Newton's method and iterative refinement, have been analyzed in, for example, [23], [7], and [103]. Numerical algorithms for handling (55), (56), (61), and a large variety of related problems are described in [8], [124], [185], [186] and a thorough survey of the Schur method, generalized eigenvalue/eigenvector extensions, and the underlying algebraic structure in terms of "Hamiltonian pencils" and "symplectic pencils" is described in [119]. Other valuable surveys on these problems are [22], [131].

Schur techniques can also be applied to Riccati differential and difference equations [116] and to nonsymmetric Riccati equations which arise in, for example, invariant imbedding methods for solving linear two-point boundary value problems [117]. Other algorithms for Riccati differential equations are surveyed in [104] and a new family of algorithms for matrix-valued differential equations, including Riccati equations, is developed in [31].

As with the linear Lyapunov and Sylvester equations, only recently have satisfactory results been obtained concerning conditioning of Riccati equations, a topic of great interest independent of what solution method is used, be it a Schur-type method or one of numerous alternatives. A number of early attempts to estimate condition numbers are compared and evaluated in [7] along with numerous examples which illustrate deficiencies in each known condition estimate. The most useful recent study of this problem is found in [93].

A very interesting class of invariant-subspace-based algorithms for solving the Riccati equation and related problems makes use of the so-called matrix sign function. These methods, which are particularly attractive for very large order problems, are described in detail in [63], [120], [29], [64], [100], and the references therein. These algorithms are based on Newton's method applied to a certain matrix equation. A new family of iterative algorithms of arbitrary order convergence has been developed in [97]. This family of algorithms can be parallelized easily and yields a viable method of solution for very high order Riccati equations. Parallel implementations of general matrix sign methods for the Riccati equation are described in [152], and further enhancements and description of a specific application involving the solution of a 556th-order Riccati equation by these methods are contained in [155]. A thorough study of the conditioning of the matrix sign function is given in [96] while scaling strategies are developed in [98].

4.9 Pole Assignment and Observer Design

The problem of designing state or output feedback for a linear system so that the resulting closed-loop system has a desired set of poles can be considered as an inverse eigenvalue problem [136], [159]. The state feedback pole assignment problem is as follows: Given a pair (A, B), one looks for a matrix F such that the eigenvalues of the matrix

$$A_F = A + BF$$

lie at specified locations or in specified regions. Many approaches have been developed for solving this problem. However, only recently has the emphasis shifted towards numerically reliable methods and consideration of the numerical sensitivity of the problem [92], [194], [161], [134], [136], [159], [137], [40]. Special cases of the pole assignment problem arise in observer design [188] and in deadbeat control for discrete-time systems (where $A+BF$ is required to be nilpotent) [187]. Part 6 contains some of the key papers in the area of pole assignment and observer design. The numerically reliable methods for pole assignment are based on the reduction of A to either a real Schur form [194], or to a Hessenberg or block Hessenberg (staircase) form [136], [159], [160], [161], [40], [137], [6]. The latter may be regarded as a numerically robust alternative to the use of a controllable or Luenberger canonical form whose computation is known to be numerically unreliable [197], [94]. For multi-input systems, the additional freedom available in the state-feedback matrix can be used for eigenvector assignment [146], and sensitivity minimization for the closed-loop poles [92], [24]. There the resulting matrix A_F is not computed directly but instead the matrices Λ and X of the decomposition

$$A_F = X\Lambda X^{-1}$$

are computed via an iterative technique. The iteration aims at minimizing the sensitivity of the placed eigenvalues λ_i or at maximizing the orthogonality of the eigenvectors x_i.

The problem of pole assignment by output feedback is a much more difficult problem both theoretically as well as computationally, and consequently there are few numerically reliable algorithms available [140], [135]. Other recent work on pole assignment has been concerned with generalized state-space or descriptor systems [55], [32], [91].

The problem of observer design for a given state-space system $\{A, B, C\}$ amounts to finding matrices T, A_K, and K such that

$$TA_K - AT = KC \tag{62}$$

whereby the spectrum of A_K is specified. Because this is an underdetermined (and nonlinear) problem in the unknown parameters of T, A_K, and K, one typically sets $T = I$ and (62) becomes then

$$A_K = A + KC$$

which is essentially a transposed pole placement problem. In this case the above techniques of pole placement automatically apply here. In reduced order design, T is nonsquare and cannot thus be put equal to the identity matrix. One can still solve (62) via a recurrence relation when assuming A_K to be in Schur form [188]. Other approaches solve this via reduced systems of equations obtained from a preliminary rank reduction [125], [10].

A more elaborate problem is that of assigning not only the eigenvalues of A_F and A_K but also their associated Jordan structure in case there are multiple eigenvalues and freedom to choose between different structures. Methods that address this problem (see, e.g., [181], [182]) are definitely more involved and their numerical reliability is also not well understood yet.

4.10 Robust Control

In the last decade, there has been very significant growth in the theory and techniques of robust control. The reprint books [46] and [47] provide an excellent coverage of these developments. It should be noted, however, that the area of robust control is still evolving and its numerical aspects have only just begun to be addressed [154]. Consequently, it would be premature to give a survey of reliable numerical algorithms in the area. However, to give a flavor of the kind of numerical and computational issues that are involved, we consider in this section two developments in robust

control which have attracted a great deal of attention. These are the H_∞ [203] and *structured singular value* [49] approaches. They have provided a powerful framework for synthesizing *robust* controllers for linear systems. The controllers are robust in the sense that they achieve desired system performance in the presence of a significant amount of uncertainty in the system.

In this section, we denote by $\mathbb{R}(s)^{n\times m}$ the set of proper real rational matrices of dimension $n \times m$. The H_∞ norm of a stable matrix $G(s) \in \mathbb{R}(s)^{n\times m}$ is defined as

$$\|G(s)\|_\infty := \sup_{\omega\in\mathbb{R}} \sigma_{max}[G(j\omega)] \tag{63}$$

where $\sigma_{max}[\cdot]$ denotes the largest singular value of a (complex) matrix. Iterative methods are available for computing this norm; see, e.g., [15], [167], [18]. In [15], a relationship is established between the singular values of $G(jw)$ and the imaginary eigenvalues of a Hamiltonian matrix obtained from a state-space realization of $G(s)$. This result is then used to develop an efficient bisection algorithm for computing the H_∞ norm of $G(s)$.

A. The H_∞ Control Problem

Consider a linear, time-invariant system described by the state-space equations

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B_1 w(t) + B_2 u(t), \\ z(t) &= C_1 x(t) + D_{11} w(t) + D_{12} u(t), \\ y(t) &= C_2 x(t) + D_{21} w(t) + D_{22} u(t), \end{aligned} \tag{64}$$

where $x(t) \in \mathbb{R}^n$ denotes the state vector; $w(t) \in \mathbb{R}^{m_1}$ is the vector of disturbance inputs; $u(t) \in \mathbb{R}^{m_2}$ is the vector of control inputs; $z(t) \in \mathbb{R}^{p_1}$ is the vector of error signals; and $y(t) \in \mathbb{R}^{p_2}$ is the vector of measured variables. The transfer function relating the inputs $\begin{bmatrix} w \\ u \end{bmatrix}$ to the outputs $\begin{bmatrix} z \\ y \end{bmatrix}$ is given by

$$G(s) := \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix} \tag{65}$$

$$= \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} + \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} (sI - A)^{-1} \begin{bmatrix} B_1 & B_2 \end{bmatrix}. \tag{66}$$

Implementing a feedback controller defined by

$$u = K(s)y \tag{67}$$

where $K(s) \in \mathbb{R}(s)^{m_2\times p_2}$, we get the closed-loop transfer matrix $T_{zw}(s) \in \mathbb{R}(s)^{p_1\times m_1}$ from the disturbance w to the regulated output z

$$T_{zw} := G_{11} + G_{12}K(I - G_{22}K)^{-1}G_{21}. \tag{68}$$

Next, define the set $\mathcal{K}$ of all internally stabilizing feedback controllers for the system in (64), i.e.,

$$\mathcal{K} := \{K(s) \in \mathbb{R}(s)^{m_2\times p_2} : T_{zw}(s) \text{ is internally stable}\}.$$

Now let $K(s) \in \mathcal{K}$ and define

$$\gamma := \|T_{zw}(s)\|_\infty. \tag{69}$$

Then the H_∞ control problem is to find a controller $K(s) \in \mathcal{K}$ which minimizes γ. The optimal value of γ is defined as

$$\gamma_{opt} := \inf_{K \in \mathcal{K}} \|T_{zw}(s)\|_\infty. \tag{70}$$

The original formulation of this problem was in an input-output setting, and the early methods for computing γ_{opt} either used an iterative search involving spectral factorization and solving the resulting Nehari problem [59], [27], [28], or computed the spectral norm of the associated Hankel plus Toeplitz operator [86]. A recent state-space formulation for computing γ_{opt} that is promising from the point of view of numerical computation is given in [48]. In this approach, the problem is formulated in terms of two algebraic Riccati equations which depend on a gain parameter γ. Then, under certain assumptions (see [48], [154] for details), it can be shown that for a controller $K(s) \in \mathcal{K}$ to exist such that $\|T_{zw}\|_\infty < \gamma$, three conditions have to be satisfied, namely, stabilizing solutions exist for the two Riccati equations, and the spectral radius of the product of the solutions is bounded by γ^2. If these conditions are satisfied for a particular value of γ, the corresponding controller $K(s)$ can be obtained from the solutions of the Riccati equations [69]. The optimal gain, γ_{opt}, is the infimum over all suboptimal values of γ such that the three conditions are satisfied.

The above approach immediately suggests a bisection-type algorithm for computing γ_{opt}. However, such an algorithm can be very slow in the neighborhood of the optimal value. To obtain speedup near the solution, a gradient approach is proposed in [153], [154]. The behavior of the Riccati solution as a function of γ is used to derive an algorithm that couples a gradient method with bisection. It has been pointed out in [154] that the Riccati equation can become ill-conditioned as the optimal value of γ is approached. It is therefore recommended in [154] that instead of computing the Riccati solutions explicitly, invariant subspaces of the associated Hamiltonian matrices should be used.

B. Structured Singular Values

Structured singular values [49] provide a systematic framework for using information about the structure of uncertainties or perturbations in control system analysis and design. The structured singular value approach therefore complements the H_∞ approach. The idea underlying the concept of the structured singular value is as follows. In a linear system with multiple independent norm-bounded perturbations, it is always possible by rearranging the system to isolate the perturbations as a single large block-diagonal perturbation Δ. Then, denoting the transfer function from the collective outputs of the perturbation to their inputs by $M(s)$, the stability problem reduces to ensuring that $\det(I + M\Delta) \neq 0$ at all frequencies and for all allowable Δ. The definition of structured singular values centers around the condition $\det(I + M\Delta) \neq 0$. A formal definition is as follows:

Let the set of allowable perturbations be denoted by $\mathcal{Q} \subseteq \mathbb{C}^{n \times n}$, and be defined as

$$\begin{aligned} \mathcal{Q} \;=\; & \{\Delta = \text{block diag}(\delta_1^r I_{k_1}, \ldots, \delta_p^r I_{k_p}, \delta_1^c I_{k_{p+1}}, \ldots, \delta_q^c I_{k_{p+q}}, \\ & \Delta_1^C, \ldots, \Delta_r^C) \;:\; \delta_i^r \in \mathbb{R},\ \delta_i^c \in \mathbb{C},\ \Delta_i^C \in \mathbb{C}^{c_i \times c_i}\}. \end{aligned} \tag{71}$$

The structured singular value of an $n \times n$ complex matrix M is then defined as

$$\mu_{\mathcal{Q}}(M) = \begin{cases} 0 \quad \text{if} \quad \det(I + M\Delta) \neq 0 \quad \text{for all } \Delta \in \mathcal{Q} \\ \left[\min_{\{\Delta \in \mathcal{Q} : \det(I + M\Delta) = 0\}} \sigma_{max}(\Delta) \right]^{-1} \quad \text{otherwise.} \end{cases} \tag{72}$$

Computing $\mu_{\mathcal{Q}}(M)$ is a difficult numerical problem. One approach, which is computationally rather demanding, is to formulate the problem as a nondifferentiable convex optimization problem

involving the maximum singular value of a matrix obtained from M. A more efficient scheme is given in [53], where it is shown that the structured singular value can be obtained by solving several smooth optimization problems that do not involve any eigenvalue or singular value computations. The computational complexity of the problem of computing $\mu_{\mathcal{Q}}(M)$ has prompted several researchers to look for bounds that are easier to compute; see, e.g., [54], [202], [9].

5 Mathematical Software

A. General Remarks

The previous sections have highlighted some topics from numerical linear algebra and their applications to numerical problems arising in systems, control, and estimation theory. Of course, these problems represent only a very small subset of numerical problems of interest in these fields but even for problems which are apparently "simple" from a mathematical point of view, the myriad of little details which constitute a sophisticated implementation become so overwhelming that the only effective means of communicating an algorithm is through its embodiment as mathematical software. Mathematical or numerical software simply means an implementation on a computing machine of an algorithm for solving a mathematical problem. Ideally, such software would be reliable, portable, and unaffected by the machine or system environment in which it is used.

The prototypical work on reliable, portable mathematical software for the standard eigenproblem was started in 1968. EISPACK, Editions I and II ([62], [171]), were an outgrowth of that work. Subsequent efforts of interest to control engineers include LINPACK [45] for linear equations and linear least squares problems, FUNPACK (Argonne) for certain function evaluations, MINPACK (Argonne) for certain optimization problems, and various ODE and PDE codes. High quality algorithms are published regularly in the *ACM Transactions on Mathematical Software*. Recently, LAPACK, the successor to LINPACK and EISPACK has been released [3]. This software has been designed to run efficiently on a wide range of machines, including vector processors, shared-memory multi-processors, and high-performance workstations.

Technology to aid in the development of mathematical software in Fortran has been assembled as a package called TOOLPACK [132]. Mechanized code development offers other advantages with respect to, for example, modifications, updates, versions, and maintenance. Excellent references on portability and other aspects of mathematical software include [35], [58], [77], [132], and [162].

Inevitably, numerical algorithms are strengthened when their mathematical software is made portable since their widespread use is greatly facilitated. Furthermore, such software has been shown to be markedly faster by factors ranging from 10 to 50 than earlier and less reliable code.

One can list many other features besides portability, reliability, and efficiency which are characteristic of "good" mathematical software. For example, one can include:

- high standards of documentation and style so as to be easily understood and used
- ease of use; ability of the user to interact with the algorithm
- consistency/compatibility/modularity in the context of a larger package or more complex problem
- error control, exception handling
- robustness with respect to unusual situations
- graceful performance degradation as problem domain boundaries are approached

- appropriate program size (a function of intended use, e.g., low accuracy, real-time applications)
- availability and maintenance
- "tricks" such as underflow-/overflow-proofing if necessary and implementation of columnwise or rowwise linear algebra [142].

Clearly, the list can go on.

What becomes apparent from the above considerations is that the evaluation of mathematical software is a highly nontrivial task; see, for example, [58]. Clearly, the quality of software is largely a function of its operational specification. It must also reflect the numerical aspects of the algorithm being implemented. The language used and the compiler (e.g., optimizing or not) used for that language will both have an enormous impact on quality—both perceived and real—as will the underlying hardware and arithmetic. Different implementations of the same algorithm can have markedly different properties and behavior. Further discussions on this subject can be found in [36], [58], [77], and [132].

Perhaps one of the most important and useful developments in mathematical software for most control engineers has been the advent of very high level systems such as Matlab, Xmath, Ctrl-C, etc. Such systems spare the engineer the drudgery of working at a detailed level with languages such as Fortran and C and they provide a large number of powerful computational "tools" (frequently through the availability of formal "toolboxes"). Of course, for many problems, the engineer must still have some knowledge of the algorithmic details embodied in such a system, and many of the papers in this volume provide the appropriate background.

B. Mathematical Software in Control

Many aspects of systems, control, and estimation theory are at the stage from which one can start the the research and design necessary to produce reliable, portable mathematical software. Certainly many of the underlying linear algebra tools (for example, in EISPACK, LINPACK, and LAPACK) are considered sufficiently reliable to be used as black—or at least gray—boxes by control engineers. Much of that theory and methodology can and has been carried over to control problems, but this really applies to only a few basic control problems. Typical examples are Riccati equations, Lyapunov equations, and certain basic state-space transformations and operations. Much of the work done in control, particularly the design and synthesis aspects, is simply not amenable to nice, "clean" algorithms and the ultimate software must have the capability to enable a dialogue between the computing machine and the control engineer, but with the latter probably still making the final engineering decisions.

6 Concluding Remarks

Several numerical issues and techniques from numerical linear algebra together with a number of important applications of these ideas have been outlined. A key question in these and other problems in systems, control, and estimation theory is what can be reliably computed and used in the presence of parameter uncertainty or structural constraints (e.g., certain "hard zeros") in the original model, and roundoff errors in the calculations. However, the ultimate goal is to solve real problems, and reliable tools (mathematical software) and experience must be available to effect real solutions or strategies. The serious interdisciplinary effort that has been under way for the last decade has significantly improved our understanding of the issues involved in reaching this goal, and has resulted in some high quality control software based on numerically reliable and well-tested

algorithms. This provides clear evidence of the fruitful symbiosis that has taken place between numerical analysis and numerical problems from control. We expect this symbiotic relationship to flourish as control engineering realizes the full potential of the computing power that is becoming more widely available in the form of multi-processing systems and high-performance workstations. However, as in other applications areas, software will continue to dominate, both as a constraint and as a vehicle for progress. Unfortunately, exceptionally high quality software is exceptionally expensive, in terms of both money and time.

Finally, it is important to note that in this paper we have focused only on dense numerical linear algebra problems in systems and control, and that we have surveyed the state of the art only in this context. Several *related topics* which have not been covered here are, e.g., *parallel algorithms, algorithms for sparse or structured matrices, optimization algorithms, ODE algorithms, approximation algorithms*, and so on. These areas are of course well established in their own right, but for applications in control there is still a lot of groundwork to be done. The main reason why we have confined ourselves to dense numerical linear algebra problems in systems and control is that, in our opinion, this area has reached a level of maturity where definitive statements and recommendations can be made about various algorithms and other developments. We hope that this has been reflected well in this survey.

References

[1] V. Adamjan, D. Arov, and M. Krein, "Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur-Takagi problem," *Mat. USSR Sbornik*, vol. 15, pp. 31–73, 1971.

[2] B.D.O. Anderson and J.B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice Hall, 1979.

[3] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorenson, *LAPACK Users' Guide*. Philadelphia, PA: SIAM, 1992.

[4] A. Antoulas, "New results on the algebraic theory of linear systems: The solution of the cover problems," *Lin. Alg. Appl.*, vol. 50, pp. 1–45, 1983.

[5] E.S. Armstrong, *ORACLS, A Design System for Linear Multivariable Control*, vol. 10 of *Control & Systems Theory Series*. New York: Dekker, 1980.

[6] M. Arnold and B.N. Datta, "An algorithm for the multi-input eigenvalue assignment problem," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 1149–1152, 1990.

[7] W.F. Arnold, III, *On the Numerical Solution of Algebraic Matrix Riccati Equations*. PhD thesis, Dept. of Elec. Eng. – Systems, Univ. of Southern California, Los Angeles, CA, December 1983.

[8] W.F. Arnold, III and A.J. Laub, "Generalized eigenproblem algorithms and software for algebraic Riccati equations," *Proceedings of the IEEE*, vol. 72, pp. 1746–1754, 1984.

[9] V. Balakrishnan, E. Feron, S. Boyd, and L. El Ghaoui, "Computing bounds for the structured singular value via an interior point algorithm," in *Proc. Amer. Contr. Conf.*, Chicago, IL, pp. 2195–2196, June 1992.

[10] J.B. Barlow, M.M. Monahemi, and D.P.O'Leary, "Constrained matrix Sylvester equations," *SIAM J. Matrix Anal. Appl.*, vol. 13, pp. 1–9, 1992.

[11] A.Y. Barraud, "Investigation autour de la fonction signe d'une matrice – application à l'équation de Riccati," *R.A.I.R.O. Automatique/Systems Analysis and Control*, vol. 13, pp. 335–368, 1979.

[12] R.H. Bartels and G.W. Stewart, "Algorithm 432: Solution of the matrix equation $AX+XB = C$," *Commun. ACM*, vol. 15, pp. 820–826, 1972.

[13] T. Beelen and P. Van Dooren, "An improved algorithm for the computation of Kronecker's canonical form of a singular pencil," *Lin. Alg. Appl.*, vol. 105, pp. 9–65, 1988.

[14] G.J. Bierman, *Factorization Methods for Discrete Sequential Estimation.* New York: Academic Press, 1977.

[15] S. Boyd, V. Balakrishnan, and P. Kabamba, "A bisection method for computing the H_∞ norm of a transfer matrix and related problems," *Math. Contr. Signals Syst.*, vol. 2, pp. 207–219, 1989.

[16] R. Brent, F. Gustavson, and D. Yun, "Fast solution of Toeplitz systems of equations and computation of Padé approximants," *Journal of Algorithms*, vol. 1, pp. 259–295, 1980.

[17] R.W. Brockett, *Finite Dimensional Linear Systems.* New York: Wiley, 1970.

[18] N.A. Bruinsma and M. Steinbuch, "A fast algorithm to compute the H_∞ norm of a transfer matrix," *Syst. Contr. Lett.*, vol. 14, pp. 287–293, 1990.

[19] A. Bultheel, "Error analysis of incoming and outgoing schemes for the trigonometric moment problem," in *Lecture notes in Mathematics* (Van Rossum and de Bruin, eds.), pp. 100–109, Berlin: Springer-Verlag, 1980.

[20] A. Bultheel, "Recursive algorithms for the matrix Padé problem," *Math. Comp.*, vol. 35, pp. 875–892, 1980.

[21] J. Bunch, "Stability of methods for solving Toeplitz systems of equations," *SIAM J. Sci. Stat. Comput.*, vol. 6, pp. 349–364, 1985.

[22] A. Bunse-Gerstner, R. Byers, and V. Mehrmann, "Numerical methods for algebraic Riccati equations," in *Proc. Workshop on the Riccati Equation in Control, Systems, and Signals (Como, Italy)* (S. Bittanti, ed.), pp. 107–116, Bologna, Italy: Pitagora Editrice, 1989.

[23] R. Byers, *Hamiltonian and Symplectic Algorithms for the Algebraic Riccati Equation.* PhD thesis, Dept. Comp. Sci., Cornell University, Ithaca, NY, 1983.

[24] R. Byers and S.G. Nash, "Approaches to robust pole assignment," *Int. J. Contr.*, vol. 49, pp. 97–117, 1989.

[25] P.R. Cappello and A.J. Laub, "Systolic computation of multivariable frequency response," *IEEE Trans. Automat. Contr.*, vol. 33, pp. 550–558, 1988.

[26] T.F. Chan, "Rank revealing QR factorizations," *Lin. Alg. Appl.*, vol. 88/89, pp. 67–82, 1987.

[27] B.C. Chang, S.S. Banda, and T.E. McQuade, "Fast iterative algorithm for H_∞ optimization problems," in *Proc. Amer. Contr. Conf.*, Minneapolis, MN, pp. 1253–1258, 1987.

[28] B.C. Chang and J.B. Pearson, "Iterative computation for minimal H_∞ norm," in *Proc. IEEE Conf. Decision Contr.*, Ft. Lauderdale, FL, pp. 1307–1310, 1985.

[29] J.P. Charlier and P. Van Dooren, "A systolic algorithm for Riccati and Lyapunov equations," *Math. Contr. Signals Syst.*, vol. 2, pp. 109–136, 1989.

[30] C. Choi and A.J. Laub, "Improving the efficiency of matrix operations in the numerical solution of large implicit systems of linear differential equations," *Int. J. Contr.*, vol. 46, pp. 991–1008, 1987.

[31] C. Choi and A.J. Laub, "Efficient matrix-valued algorithms for solving stiff Riccati differential equations," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 770–776, 1990. (See also *Proc. 1989 IEEE Conf. Decision Contr.*, pp. 885-887).

[32] K.-W.E. Chu, "A controllability condensed form and a state feedback pole assignment algorithm for descriptor systems," *IEEE Trans. Automat. Contr.*, vol. 33, pp. 366–370, 1988.

[33] A.K. Cline, C.B. Moler, G.W. Stewart, and J.H. Wilkinson, "An estimate for the condition number of a matrix," *SIAM J. Numer. Anal.*, vol. 16, pp. 368–375, 1979.

[34] T.P. Coleman and C.F. Van Loan, *Handbook for Matrix Computations.* Philadelphia, PA: SIAM, 1988.

[35] W. Cowell, *Portability of Numerical Software.* New York: Springer-Verlag, 1977. Oak Brook, 1976, (Lecture Notes Comput Sci., Vol. 57).

[36] H. Crowder, R.S. Dembo, and J.M. Mulvey, "On reporting computational experiments with mathematical software," *ACM Trans. Math. Software*, vol. 5, pp. 193–203, 1979.

[37] G. Cybenko, "The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations," *SIAM J. Sci. Stat. Comput.*, vol. 1, pp. 303–319, 1980.

[38] G. Cybenko, "The numerical stability of the lattice algorithm for least squares linear prediction problems," *BIT*, vol. 24, pp. 441–455, 1984.

[39] G. Dahlquist and A. Bjorck, *Numerical Methods.* Englewood Cliffs, NJ: Prentice Hall, 1974.

[40] B.N. Datta, "An algorithm to assign eigenvalues in a Hessenberg matrix: Single-input case," *IEEE Trans. Automat. Contr.*, vol. AC-32, pp. 414–417, 1987.

[41] L.S. de Jong, "Towards a formal definition of numerical stability," *Numer. Math.*, vol. 28, pp. 211–220, 1977.

[42] L.S. de Jong, "Numerical aspects of the recursive realization algorithm," *SIAM J. Contr. Optimiz.*, vol. 16, pp. 646–659, 1978.

[43] J. Demmel and B. Kågström, "Computing stable eigen-decompositions of matrix pencils $A - \lambda B$," *Lin. Alg. Appl.*, vol. 88/89, pp. 139–186, 1987.

[44] J. Demmel and W. Kahan, "Accurate singular values of bidiagonal matrices," *SIAM J. Sci. Stat. Comput.*, vol. 11, pp. 873–912, 1990.

[45] J.J. Dongarra, J.R. Bunch, C.B. Moler, and G.W. Stewart, *LINPACK Users' Guide*. Philadelphia, PA: SIAM, 1979.

[46] P. Dorato, ed., *Robust Control*. Selected Reprint Series, New York: IEEE Press, 1987.

[47] P. Dorato and R.K. Yedavalli, eds., *Recent Advances in Robust Control*. Selected Reprint Series, New York: IEEE Press, 1990.

[48] J. Doyle, K. Glover, P.P. Khargonekar, and B.A. Francis, "State-space solutions to standard H_2 and H_∞ control problems," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 831–847, 1989.

[49] J.C. Doyle, "Analysis of feedback systems with structured uncertainties," *IEE Proc., Pt. D*, vol. 129, pp. 242–250, 1982.

[50] J.C. Doyle and G. Stein, "Multivariable feedback: Concepts for a classical/modern synthesis," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 4–16, 1981.

[51] A. Emami-Naeini and P. Van Dooren, "Computation of zeros of linear multivariable systems," *Automatica*, vol. 18, pp. 415–430, 1982.

[52] W. Enright, "On the efficient and reliable numerical solution of large linear systems of ODE's," *IEEE Trans. Automat. Contr.*, vol. AC-24, pp. 905–908, 1979.

[53] M.K.H. Fan and A.L. Tits, "Characterization and efficient computation of the structured singular value," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 734–743, 1986.

[54] M.K.H. Fan, A.L. Tits, and J.C. Doyle, "Robustness in the presence of mixed parametric uncertainty and unmodeled dynamics," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 25–38, 1991.

[55] L.R. Fletcher, J. Kautsky, and N.K. Nichols, "Eigenstructure assignment in descriptor systems," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 1138–1141, 1986.

[56] G.E. Forsythe, M.A. Malcolm, and C.B. Moler, *Computer Methods for Mathematical Computations*. Englewood Cliffs, NJ: Prentice Hall, 1977.

[57] G.E. Forsythe and C.B. Moler, *Computer Solution of Linear Algebraic Systems*. Englewood Cliffs, NJ: Prentice Hall, 1967.

[58] L.D. Fosdick, ed., *Performance Evaluation of Numerical Software*. Amsterdam: North-Holland, 1979.

[59] B.A. Francis, *A Course in H_∞ Control Theory*. New York: Springer-Verlag, 1987.

[60] J.G.F. Francis, "The QR transformation I," *Comput. J.*, vol. 4, pp. 265–271, 1961.

[61] J.G.F. Francis, "The QR transformation II," *Comput. J.*, vol. 4, pp. 332–345, 1962.

[62] B.S. Garbow, J.M. Boyle, J.J. Dongarra, and C.B. Moler, *Matrix Eigensystem Routines – EISPACK Guide Extension*, vol. 51 of *Lecture Notes in Computer Science*. New York: Springer-Verlag, 1977.

[63] J.D. Gardiner and A.J. Laub, "A generalization of the matrix-sign-function solution for algebraic Riccati equations," *Int. J. Contr.*, vol. 44, pp. 823–832, 1986. (See also *Proc. 1985 IEEE Conf. Decision Contr.*, pp. 1233-1235).

[64] J.D. Gardiner and A.J. Laub, "Parallel algorithms for algebraic Riccati equations," *Int. J. Contr.*, vol. 54, pp. 1317–1333, 1991.

[65] J.D. Gardiner, A.J. Laub, J.J. Amato, and C.B. Moler, "Solution of the Sylvester matrix equation $AXB^T + CXD^T = E$," *ACM Trans. Math. Software*, vol. 18, pp. 223–231, 1992.

[66] J.D. Gardiner, M.R. Wette, A.J. Laub, J.J. Amato, and C.B. Moler, "Algorithm 705: A Fortran-77 software package for solving the Sylvester matrix equation $AXB^T+CXD^T = E$," *ACM Trans. Math. Software*, vol. 18, pp. 232–238, 1992.

[67] W.M. Gentleman and H.T. Kung, "Matrix triangularization by systolic arrays," in *Proc. SPIE Symp.: Real-Time Signal Processing IV*, vol. 298, pp. 19–26, 1981.

[68] K. Glover, "All optimal Hankel-Norm approximations of linear mutivariable systems and their L_∞ error bounds," *Int. J. Contr.*, vol. 39, pp. 1115–1193, 1984.

[69] K. Glover and J. Doyle, "State-space formulas for all stabilizing controllers that satisfy an H_∞ norm bound and relations to risk sensitivity," *Syst. Contr. Lett.*, vol. 11, pp. 167–172, 1988.

[70] 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.

[71] G.H. Golub, S. Nash, and C.F. Van Loan, "A Hessenberg-Schur method for the problem $AX + XB = C$," *IEEE Trans. Automat. Contr.*, vol. AC-24, pp. 909–913, 1979.

[72] G.H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numer. Math*, vol. 14, pp. 403–420, 1970.

[73] G.H. Golub and C.F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins University Press, Second Edition, 1989. (First Edition: 1983).

[74] G.H. Golub and J.H. Wilkinson, "Ill-conditioned eigensystems and the computation of the Jordan canonical form," *SIAM Review*, vol. 18, pp. 578–619, 1976.

[75] S.J. Hammarling, "Numerical solution of the stable, non-negative definite Lyapunov equation," *IMA J. Numer. Anal.*, vol. 2, pp. 303–323, 1982.

[76] D. Heller, "A survey of parallel algorithms in numerical linear algebra," *SIAM Review*, vol. 20, pp. 740–777, 1978.

[77] M.A. Hennel and L.M. Delves, eds., *Production and Assessment of Numerical Software*. New York: Academic Press, 1980.

[78] P. Henrici, "Fast Fourier methods in computational complex analysis," *SIAM Review*, vol. 21, pp. 481–527, 1979.

[79] P. Henrici, *Essentials of Numerical Analysis*. New York: Wiley, 1982.

[80] G.A. Hewer and C.S. Kenney, "The sensitivity of the stable Lyapunov equation," *SIAM J. Contr. Optimiz.*, vol. 26, pp. 321–344, 1988.

[81] A.S. Householder, *The Theory of Matrices in Numerical Analysis*. New York: Blaisdell, 1964.

[82] F. Itakura and S. Saito, "Digital filtering techniques for speech analysis and synthesis," in *Conf. Rec., 7th Int. Congress on Acoustics*, Budapest, Hungary, pp. 261–264, 1971.

[83] M. Jenkins and J. Traub, "A three-stage algorithm for real polynomials using quadratic iteration," *SIAM J. Numer. Anal.*, vol. 7, pp. 545–566, 1970.

[84] J. Jezek and V. Kucera, "Efficient algorithm for spectral factorization," in *Proceedings 9th IFAC Congress*, Budapest, Hungary, pp. 257–262, July 1984.

[85] R.L. Johnston, *Numerical Methods: A Software Approach*. New York: Wiley, 1982.

[86] J.C. Juang and E.A. Jonckheere, "On computing the spectral radius of the Hankel plus Toeplitz operator," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 1053–1059, 1988.

[87] B. Kågström, "Bounds and perturbation bounds for the matrix exponential," *BIT*, vol. 17, pp. 39–57, 1977.

[88] B. Kågström and A. Ruhe, eds., *Matrix Pencils (Proceedings, Pite Havsbad, 1982)*, vol. 973 of *Lecture Notes in Mathematics*. Berlin: Springer-Verlag, 1983.

[89] W. Kahan, "Conserving confluence curbs ill-condition," Tech. Rep., Dept. Comp. Sci., University of California, Berkeley, CA, 1972.

[90] D. Kahaner, C. Moler, and S. Nash, *Numerical Methods and Software*. Englewood Cliffs, NJ: Prentice Hall, 1989.

[91] J. Kautsky, N.K. Nichols, and K.-W.E. Chu, "Robust pole assignment in singular control systems," *Lin. Alg. Appl.*, vol. 121, pp. 9–37, 1989.

[92] J. Kautsky, N.K. Nichols, and P. Van Dooren, "Robust pole assignment in linear state feedback," *Int. J. Contr.*, vol. 41, pp. 1129–1155, 1985.

[93] C.S. Kenney and G. Hewer, "The sensitivity of the algebraic and differential Riccati equations," *SIAM J. Contr. Optimiz.*, vol. 28, pp. 50–69, 1990.

[94] C.S. Kenney and A.J. Laub, "Controllability and stability radii for companion form systems," *Math. Contr. Signals Syst.*, vol. 1, pp. 239–256, 1988.

[95] C.S. Kenney and A.J. Laub, "Condition estimates for matrix functions," *SIAM J. Matrix Anal. Appl.*, vol. 10, pp. 191–209, 1989.

[96] C.S. Kenney and A.J. Laub, "Polar decomposition and matrix sign function condition estimates," *SIAM J. Sci. Stat. Comput.*, vol. 12, pp. 488–504, 1991.

[97] C.S. Kenney and A.J. Laub, "Rational iterative methods for the matrix sign function," *SIAM J. Matrix Anal. Appl.*, vol. 12, pp. 273–291, 1991.

[98] C.S. Kenney and A.J. Laub, "On scaling Newton's method for polar decomposition and the matrix sign function," *SIAM J. Matrix Anal. Appl.*, vol. 13, pp. 688–706, 1992. See also *Proc. 1990 Amer. Contr. Conf.*, pp. 2560-2564.

[99] C.S. Kenney and A.J. Laub, "Small-sample statistical condition estimates for general matrix functions," *SIAM J. Sci. Stat. Comput.*, vol. 14, 1993. To appear.

[100] C.S. Kenney, A.J. Laub, and P.M. Papadopoulos, "A Newton-squaring algorithm for finding the negative invariant subspace of a matrix," *IEEE Trans. Automat. Contr.*, vol. 38, 1993. To appear.

[101] C.S. Kenney, A.J. Laub, and S.C. Stubberud, "Frequency response computation via rational interpolation," *IEEE Trans. Automat. Contr.*, vol. 38, 1993. To appear.

[102] C.S. Kenney, A.J. Laub, and M. Wette, "A stability-enhancing scaling procedure for Schur-Riccati solvers," *Syst. Contr. Lett.*, vol. 12, pp. 241–250, 1989.

[103] C.S. Kenney, A.J. Laub, and M. Wette, "Error bounds for Newton refinement of solutions to algebraic Riccati equations," *Math. Contr. Signals Syst.*, vol. 3, pp. 211–224, 1990.

[104] C.S. Kenney and R.B. Leipnik, "Numerical integration of the differential matrix Riccati equation," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 962–970, 1985.

[105] V.C. Klema and A.J. Laub, "The singular value decomposition: Its computation and some applications," *IEEE Trans. Automat. Contr.*, vol. AC-25, pp. 164–176, 1980.

[106] M.M. Konstantinov, P.Hr. Petkov, and N.D. Christov, "Invariants and canonical forms for linear multivariable systems under the action of orthogonal transformation groups," *Kybernetika*, vol. 17, pp. 413–424, 1981.

[107] V. Kucera, *Discrete Linear Control: The Polynomial Equation Approach.* Chichester, England: Wiley, 1979.

[108] S. Kung, "A new identification and model reduction algorithm via singular value decompositions," in *Proc. 12th Asilomar Conf. Circuits, Syst. Comp.*, pp. 705–714, Nov. 1978.

[109] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems.* New York: Wiley-Interscience, 1972.

[110] A.J. Laub, "A Schur method for solving algebraic Riccati equations," Tech. Rep. LIDS-R-859, MIT, Lab. for Inform. and Decis. Syst., Cambridge, MA, October 1978. (Including software).

[111] A.J. Laub, "A Schur method for solving algebraic Riccati equations," *IEEE Trans. Automat. Contr.*, vol. AC-24, pp. 913–921, 1979. (See also *Proc. IEEE Conf. Decision Contr. (Jan. 1979)*, pp. 60-65).

[112] A.J. Laub, "On computing "balancing" transformations," in *Proc. Joint. Autom. Contr. Conf.*, San Francisco, CA, pp. FA8–E, 1980.

[113] A.J. Laub, "Robust stability of linear systems – some computational considerations," in *Information Linkage Between Applied Mathematics and Industry II* (A.L. Schoenstadt, F.D. Faulkner, R. Franke, and I.B. Russak, eds.), pp. 57–84, New York: Academic Press, 1980.

[114] A.J. Laub, "Survey of computational methods in control theory," in *Electric Power Problems: The Mathematical Challenge* (A.M. Erisman, K.W. Neves, and M.H. Dwarakanath, eds.), pp. 231–260, Philadelphia, PA: SIAM, 1980.

[115] A.J. Laub, "Efficient multivariable frequency response computations," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 407–408, 1981.

[116] A.J. Laub, "Schur techniques for Riccati differential equations," in *Feedback Control of Linear and Nonlinear Systems* (D. Hinrichsen and A. Isidori, eds.), pp. 165–174, New York: Springer-Verlag, 1982.

[117] A.J. Laub, "Schur techniques in invariant imbedding methods for solving two-point boundary value problems," in *Proc. IEEE Conf. Decision Contr.*, Orlando, FL, pp. 55–61, December 1982.

[118] A.J. Laub, "Algorithm 640: Efficient calculation of frequency response matrices from state space models," *ACM Trans. Math. Software*, vol. 12, pp. 26–33, 1986.

[119] A.J. Laub, "Invariant subspace methods for the numerical solution of Riccati equations," in *The Riccati Equation* (S. Bittanti, A.J. Laub, and J.C. Willems, eds.), pp. 163–196, Berlin: Springer-Verlag, 1991.

[120] A.J. Laub and J.D. Gardiner, "Hypercube implementation of some parallel algorithms in control," in *Advanced Computing Concepts and Techniques in Control Engineering* (M.J. Denham and A.J. Laub, eds.), pp. 361–390, Berlin: Springer-Verlag, 1988.

[121] A.J. Laub, M.T. Heath, C.C. Paige, and R.C. Ward, "Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms," *IEEE Trans. Automat. Contr.*, vol. AC-32, pp. 115–122, 1987.

[122] A.J. Laub and B.C. Moore, "Calculation of transmission zeros using QZ techniques," *Automatica*, vol. 14, pp. 557–566, 1978.

[123] C.L. Lawson and R.J. Hanson, *Solving Least Squares Problems*. Englewood Cliffs, NJ: Prentice Hall, 1974.

[124] K.H. Lee, *Generalized Eigenproblem Structures and Solution Methods for Riccati Equations*. PhD thesis, Dept. of Elec. Eng. – Systems, Univ. of Southern California, Los Angeles, CA, January 1983.

[125] F.L. Lewis, "Geometric techniques for observer design in singular systems," *Automatica*, vol. 26, pp. 411–415, 1990.

[126] L. Ljung, *System Identification: Theory for the User*. Englewood Cliffs, NJ: Prentice Hall, 1987.

[127] R.E. Lord, J.S. Kowalik, and S.P. Kumar, "Solving linear algebraic equations on a MIMD computer," in *Proc. 1980 Int. Conf. on Parallel Proc.*, pp. 205–210, 1980.

[128] F.T. Luk, "A rotation method for computing the QR factorization," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 452–459, 1986.

[129] J. Makhoul, "Linear prediction : A tutorial review," *Proceedings of the IEEE*, vol. 63, pp. 561–580, 1975.

[130] R.S. Martin and J.H. Wilkinson, "Similarity reduction of a general matrix to Hessenberg form," *Numer. Math.*, vol. 12, pp. 349–368, 1968.

[131] V. Mehrmann, *The Autonomous Linear Quadratic Control Problem*, vol. 163 of *Lecture Notes in Control and Information Sciences*. New York: Springer-Verlag, 1991.

[132] P. Messina and A. Murli, eds., *Problems and Methodologies in Mathematical Software Production.* New York: Springer-Verlag, 1982. (Lecture Notes in Comput. Sci., No. 142).

[133] W. Miller and C. Wrathall, *Software for Roundoff Analysis of Matrix Algorithms.* New York: Academic Press, 1980.

[134] G. Miminis and C.C. Paige, "An algorithm for pole assignment of time invariant linear systems," *Int. J. Contr.*, vol. 35, pp. 341–354, 1982.

[135] G.S. Miminis, "Using deflation in the pole assignment problem with output feedback," in *Proc. 3rd Conf. on Aerospace Computational Contr.*, Oxnard, CA, pp. 140–154, 1989.

[136] G.S. Miminis and C.C. Paige, "An algorithm for pole assignment of time-invariant multi-input linear systems," in *Proc. IEEE Conf. Decision Contr.*, Orlando, FL, pp. 62–67, December 1982.

[137] G.S. Miminis and C.C. Paige, "A direct algorithm for pole assignment of time-invariant multi-input linear systems using state feedback," *Automatica*, vol. 24, pp. 343–356, 1988.

[138] P. Misra, "Hessenberg-triangular reduction and transfer function matrices of singular systems," *IEEE Trans. Circuits Syst.*, vol. CAS-36, pp. 907–912, 1989.

[139] P. Misra and R.V. Patel, "A determinant identity and its application in evaluating frequency response matrices," *SIAM J. Matrix Anal. Appl.*, vol. 9, pp. 248–255, 1988.

[140] P. Misra and R.V. Patel, "Numerical algorithms for eigenvalue assignment by constant and dynamic output feedback," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 579–588, 1989.

[141] J.J. Modi, *Parallel Algorithms and Matrix Computations.* Oxford, England: Oxford University Press, 1988.

[142] C.B. Moler, "Matrix computations with Fortran and paging," *Commun. ACM*, vol. 15, pp. 268–270, 1972.

[143] C.B. Moler and G.W. Stewart, "An algorithm for generalized matrix eigenvalue problems," *SIAM J. Numer. Anal.*, vol. 10, pp. 241–256, 1973.

[144] C.B. Moler and C.F. Van Loan, "Nineteen dubious ways to compute the exponential of a matrix," *SIAM Review*, vol. 20, pp. 801–836, 1978.

[145] M. Moonen, B. De Moor, L. Vandenberghe, and J. Vandewalle, "On- and off-line identification of linear state-space models," *Int. J. Contr.*, vol. 49, pp. 219–232, 1989.

[146] B.C. Moore, "On the flexibility offered by state feedback in multivariable systems beyond closed-loop eigenvalue assignment," *IEEE Trans. Automat. Contr.*, vol. AC-21, pp. 689–692, 1976.

[147] B.C. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 17–31, 1981.

[148] B.C. Moore and A.J. Laub, "Computation of supremal (A,B)-invariant and controllability subspaces," *IEEE Trans. Automat. Contr.*, vol. AC-23, pp. 783–792, 1978.

[149] C. Mullis and R. Roberts, "Synthesis of minimum roundoff noise fixed point digital filters," *IEEE Trans. Circuits Syst.*, vol. CAS-23, pp. 551–562, 1976.

[150] A.M. Ostrowski, "On the spectrum of a one-parametric family of matrices," *Journal für die Reine und Angewandte Mathematik*, vol. 193, pp. 143–160, 1954.

[151] C.C. Paige, "Properties of numerical algorithms related to computing controllability," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 130–138, 1981.

[152] P. Pandey, C.S. Kenney, and A.J. Laub, "A parallel algorithm for the matrix sign function," *Int. J. High Speed Computing*, vol. 2, pp. 181–191, 1990.

[153] P. Pandey, C.S. Kenney, A. Packard, and A.J. Laub, "A gradient method for computing the optimal H_∞ norm," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 887–890, 1991.

[154] P. Pandey and A.J. Laub, "Numerical issues in robust control design techniques," in *Advances in Control and Dynamic Systems, Vols. 49-52* (C.T. Leondes, ed.), San Diego, CA: Academic Press, 1992.

[155] P.M. Papadopoulos, A.J. Laub, C.S. Kenney, P. Pandey, G. Ianculescu, and J. Ly, "Optimal control study for the Space Station solar dynamic power module," in *Proc. IEEE Conf. Decision Contr.*, Brighton, England, pp. 2224–2229, 1991.

[156] T. Pappas, A.J. Laub, and N.R. Sandell, "On the numerical solution of the discrete-time algebraic Riccati equation," *IEEE Trans. Automat. Contr.*, vol. AC-25, pp. 631–641, 1980.

[157] B.N. Parlett, *The Symmetric Eigenvalue Problem*. Englewood Cliffs, NJ: Prentice Hall, 1980.

[158] R.V. Patel, "Computation of minimal order state-space realizations and observability indices using orthogonal transformations," *Int. J. Contr.*, vol. 33, pp. 227–246, 1981.

[159] R.V. Patel and P. Misra, "Numerical algorithms for eigenvalue assignment by state feedback," *Proceedings of the IEEE*, vol. 72, pp. 1755–1764, 1984.

[160] P.Hr. Petkov, N.D. Christov, and M.M. Konstantinov, "A computational algorithm for pole assignment of linear single-input systems," *IEEE Trans. Automat. Contr.*, vol. AC-29, pp. 1045–1048, 1984.

[161] P.Hr. Petkov, N.D. Christov, and M.M. Konstantinov, "A computational algorithm for pole assignment of linear multi-input systems," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 1044–1047, 1986.

[162] J.K. Reid, ed., *The Relationship Between Numerical Computation and Programming Languages*. Amsterdam: North-Holland, 1982.

[163] J.R. Rice, "A theory of condition," *SIAM J. Numer. Anal.*, vol. 3, pp. 287–310, 1966.

[164] J.R. Rice, *Matrix Computations and Mathematical Software*. New York: McGraw-Hill, 1981.

[165] J.R. Rice, *Numerical Methods, Software, and Analysis*. New York: McGraw-Hill, 1983.

[166] J. Rissanen, "Algorithm for triangular decomposition of block Hankel and Toeplitz matrices with application to factoring positive matrix polynomials," *Math. Comp.*, vol. 27, pp. 147–154, 1973.

[167] G. Robel, "On computing the infinity norm," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 882–884, 1989.

[168] M.G. Safonov, A.J. Laub, and G.L. Hartmann, "Feedback properties of multivariable systems: The role and use of the return difference matrix," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 47–65, 1981.

[169] A.H. Sameh and D.J. Kuck, "On stable parallel linear system solvers," *J. Assoc. Comput. Mach.*, vol. 25, pp. 81–91, 1978.

[170] R.F. Sincovec, A. Erisman, E. Yip, and M. Epton, "Analysis of descriptor systems using numerical algorithms," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 139–147, 1981.

[171] B.T. Smith, J.M. Boyle, J.J. Dongarra, B.S. Garbow, Y. Ikebe, V.C. Klema, and C.B. Moler, *Matrix Eigensystem Routines – EISPACK Guide*, vol. 6 of *Lecture Notes in Computer Science*. New York: Springer-Verlag, Second Edition, 1976.

[172] Special Issue, "Linear Multivariable Control Systems," *IEEE Trans. Automat. Contr.*, vol. AC-26, Feb. 1981.

[173] G.W. Stewart, "Error and perturbation bounds for subspaces associated with certain eigenvalue problems," *SIAM Review*, vol. 15, pp. 727–764, 1973.

[174] G.W. Stewart, *Introduction to Matrix Computations*. New York: Academic Press, 1973.

[175] G.W. Stewart, "Algorithm 506 – HQR3 and EXCHNG: Fortran subroutines for calculating and ordering the eigenvalues of a real upper Hessenberg matrix," *ACM Trans. Math. Software*, vol. 2, pp. 275–280, 1976.

[176] G.W. Stewart, "On the perturbation of pseudo-inverses, projections, and linear least squares," *SIAM Rev.*, vol. 19, pp. 634–662, 1977.

[177] G.W. Stewart, "Rank degeneracy," *SIAM J. Sci. Stat. Comput.*, vol. 5, pp. 403–413, 1984.

[178] G.W. Stewart and Ji-guang Sun, *Matrix Perturbation Theory*. New York: Academic Press, 1990.

[179] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*. New York: Springer-Verlag, 1980.

[180] S.C. Stubberud, A.J. Laub, and C.S. Kenney, "Computation of frequency response of descriptor systems by rational interpolation," in *Advances in Control and Dynamic Systems, Vols. 49-52* (C.T. Leondes, ed.), San Diego, CA: Academic Press, 1992.

[181] C.-C. Tsui, "An algorithm for the design of multi-functional observers," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 89–93, 1985.

[182] C.-C. Tsui, "An algorithm for computing state feedback in multi-input linear systems," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 243–246, 1986.

[183] P. Van Dooren, "The computation of Kronecker's canonical form of a singular pencil," *Lin. Alg. Appl.*, vol. 27, pp. 103–141, 1979.

[184] P. Van Dooren, "The generalized eigenstructure problem in linear system theory," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 111–129, 1981.

[185] P. Van Dooren, "A generalized eigenvalue approach for solving Riccati equations," *SIAM J. Sci. Stat. Comput.*, vol. 2, pp. 121–135, 1981.

[186] P. Van Dooren, "Algorithm 590 – DSUBSP and EXCHQZ: Fortran subroutines for computing deflating subspaces with specified spectrum," *ACM Trans. Math. Software*, vol. 8, pp. 376–382, 1982.

[187] P Van Dooren, "Deadbeat control : A special inverse eigenvalue problem," *BIT*, vol. 24, pp. 681–699, 1984.

[188] P. Van Dooren, "Reduced order observers: A new algorithm and proof," *Syst. Contr. Lett.*, vol. 4, pp. 243–251, 1984.

[189] P. Van Dooren and P. Dewilde, "The eigenstructure of an arbitrary polynomial matrix: Computational aspects," *Lin. Alg. Appl.*, vol. 50, pp. 545–579, 1983.

[190] P. Van Dooren and M. Verhaegen, "On the use of unitary state-space transformations," in *Linear Algebra and its Role in Systems Theory* (B.N. Datta, ed.), pp. 447–463, Providence, RI: AMS Contemporary Mathematics Series, vol. 47, 1985.

[191] P. Van Dooren and M. Verhaegen, "Condensed forms for efficient time-invariant Kalman filtering," *SIAM J. Sci. Stat. Comput.*, vol. 9, pp. 516–530, 1988.

[192] C. Van Loan, "Computing integrals involving the matrix exponential," *IEEE Trans. Automat. Contr.*, vol. AC-23, pp. 395–404, 1978.

[193] C.F. Van Loan, "The sensitivity of the matrix exponential," *SIAM J. Numer. Anal.*, vol. 14, pp. 971–981, 1977.

[194] A. Varga, "A Schur method for pole assignment," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 517–519, 1981.

[195] A. Varga, "On numerical simulation of linear continuous control systems," in *Proc. Simulation '83 Symp.*, Prague, Czechoslovakia, July 1983.

[196] Z. Vostry, "New algorithm for polynomial spectral factorization with quadratic convergence," *Kybernetika*, vol. 11, pp. 415–422, 1975.

[197] J.H. Wilkinson, *Rounding Errors in Algebraic Processes*. Englewood Cliffs, NJ: Prentice Hall, 1963.

[198] J.H. Wilkinson, *The Algebraic Eigenvalue Problem*. Oxford, England: Oxford University Press, 1965.

[199] J.H. Wilkinson and C. Reinsch, *Handbook for Automatic Computation Vol. II, Linear Algebra*. New York: Springer-Verlag, 1971.

[200] A. Willsky, *Digital Signal Processing and Control and Estimation Theory: Points of Tangency, Areas of Intersection and Parallel Direction*. Cambridge, MA: MIT Press, 1979.

[201] W.M. Wonham, *Linear Multivariable Control: A Geometric Approach.* New York: Springer-Verlag, Second Edition, 1979. (Third Edition: 1985).

[202] P.M. Young, M.P. Newlin, and J.C. Doyle, "Practical computation of the mixed μ problem," in *Proc. Amer. Contr. Conf.*, Chicago, IL, pp. 2190–2194, June 1992.

[203] G. Zames, "Feedback and optimal sensitivity: Model reference transformations, multiplicative semi-norms, and approximate inverses," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 301–320, 1981.

[204] H. Zeiger and A. McEwen, "Approximate linear realizations of given dimension via Ho's algorithm," *IEEE Trans. Automat. Contr.*, vol. AC-19, p. 153, 1974.