Single-input Eigenvalue Assignment Algorithms: A Close-Look

by

Mark Arnold*

and

Biswa Nath Datta**

*The author is with the Department of Mathematical Sciences, University of Arkansas, Fayetteville, Arkansas. (**E-mail**: arnold@mysong.uark.edu).

The author is with Department of Mathematical Sciences, Northern Illinois University, DeKalb, Illinois 60115 (E-mail**: dattab@math.niu.edu). The research of this author was supported by NSF grant under contract DMS-9212629.).

ABSTRACT

A close-look is given at the single-input eigenvalue assignment methods. Several previously known backward stable QR algorithms are tied together in a common framework of which each is a special case, and their connection to an explicit expression for the feedback vector is exposed. A simple new algorithm is presented and its backward stability is established by round-off-error analysis. The differences between this new algorithm and the other QR algorithms are discussed. Also, the round-off error analysis of a simple recursive algorithm for the problem (Datta (1987)) is presented. The analysis shows that the latter is reliable, and the reliability can be determined during the execution of the algorithm rather cheaply. Finally, some numerical experiments comparing some of the methods are reported.

1 Introduction.

Given a controllable pair of matrices (A, B) and a set $\Omega = \{\lambda_1, ..., \lambda_n\}$, closed under complex conjugation, the well-known eigenvalue assignment problem in control theory is the problem of finding a matrix F such that $A + BF$ has the spectrum Ω (see Chen (1984), Kailath (1980), Szidarovszky and Bahill (1991), etc).

Because of its importance, the problem has been very well-studied in both mathematics and control literatures. Many methods exist: single-input and multi-input (Arnold and Datta (1990), Bhattacharyya and DeSouza (1982), Bru et al (1994a, 1994b), Datta (1987), Miminis and Paige (1981, 1988), Patel and Misra (1981), Petkov, Christov, and Konstantinov (1984, 1986), Tsui (1986), Varga (1981), etc.); robust eigenvalue assignment (Kautsky, Nichols, and Van Dooren (1985)); partial eigenvalue assignment (Datta and Saad (1991), Saad (1988)); parallel algorithms (Arnold (1992), Bru et al (1994c), Datta (1989), Datta (1991), Coutinho et al (1995), Datta and Datta (1986), etc.); and methods for second-order systems (Datta, Elhay, and Ram (1995), Chu and Datta (1995)). The backward-stability of some of these algorithms have been established by round off error analysis (Cox and Moss (1989 and 1992), Miminis and Paige (1988)).

We take another look at the single-input methods in this paper.

In theory all the single-input algorithms produce the same solution (see Wonham (1979)). It is therefore natural to explore the relationships between these methods. We relate the QR methods of Miminis and Paige (1981), Patel and Misra (1984), and Petkov-Christov and Konstantinov (1984) under one umbrella and then relate the recursive algorithm of Datta (1987) to these results. Specifically, we prove a result that shows that all these methods are connected by a simple property of QR iteration and the explicit closed form solution of the single-input eigenvalue assignment problem that can be obtained easily from the recursive algorithm.

These results do not seem to have appeared in the literature before. The relationship allows us to present the QR algorithms in an **unified framework** through RQ factorizations of deflated matrices at each step. The unified RQ reformulations of these algorithms are easier to understand and implement than the original algorithms.

We also present a new algorithm based on the RQ formulation of the single-input recursive algorithm. We show how this new algorithm differs from other QR algorithms, and establish backward numerical stability of the algorithm through round-off error analysis. In the course of proving backward stability of this algorithm, we prove that any single-input eigenvalue assignment algorithm is backward stable if the associated Hessenberg-algorithm is backward stable.

Finally, we present a detailed round-off error analysis of the single-input recursive algorithm, which is most efficient, and almost trivial to implement, but is assumed to be numerically suspect. Our analysis shows that the stability of the method can not be guaranteed in general, but the method is reliable in the sense that we can get an indication, as the method is executed, when the results are suspect, and the indication can be obtained rather cheaply.

The organization of this paper is as follows:

In Section 2 we present an unified RQ reformation of the three QR methods.

In Section 3 we establish a relationship between these QR methods and the recursive algorithm.

In Section 4, we present a new RQ-based algorithm and discuss the differences of this new algorithm with the others.

In Section 5 we present the round-off analyses of the proposed algorithm and that of the recursive algorithm.

Finally, in Section 6 we present some numerical experiments comparing some of the methods.

2 Hessenberg Eigenvalue Assignment

The methods to be discussed in this section have the following basic structure: the pair (A, b) is first transformed to a controller-Hessenberg form; the desired feedback is then computed for the reduced problem, and finally the solution to the original problem is retrieved from the solution of the reduced problem. Recall that for single-input systems, if the Hessenberg matrix in the controller-Hessenberg form is unreduced, then the system is controllable. The above can be summarized in the following algorithm template:

Algorithm 2.1 A General Single Input Algorithm

Input: $A \in \boldsymbol{R}^{n\times n}, b \in \boldsymbol{R}^n$, and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: $f \in \boldsymbol{R}^n$ such that $\lambda(A - bf^t) = \Omega$

Step 1 Reduce the pair (A, b) to controller-Hessenberg form $(H, \beta e_1) = (PAP^t, Pb)$

Step 2 Compute $k \in \boldsymbol{R}^n$ such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 3 Compute $f = P^t k$

If, in *step* 1 it is decided that the system is uncontrollable (i.e. if H is reduced or $\beta = 0$) and if those eigenvalues of A which cannot be moved (called *uncontrollable modes*) do not belong to Ω, then we must stop with failure: Ω is unassignable. If the uncontrollable modes are contained in Ω, then we go to *step* 2 with the controllable part of H and the subset of Ω that remains to be assigned. Since Ω is closed with respect to complex conjugation, then f will be real.

We note here that the (orthogonal) matrix P determined by the reduction must be saved for use in *step 3*. **Also note that steps 1 and 3 are individually backward stable operations.** We will show in Section 5 that a method that is backward stable for Hessenberg eigenvalue assignment problem (Step 2) will be backward stable overall.

Remarks: Several remarks on Algorithm 2.1 are in order.

First. The reduction to Controller-Hessenberg form can be achieved in a numerically stable way using a stair-case algorithm (see, Boley (1981), Paige (1980) and Van Dooren and Verhaegen (1985), etc.).

Second. Step 1 and Step 3 are the same in all the eigenvalue-assignment methods to be discussed in this paper.

The different algorithms differ in the way Step 2 is implemented. We will present below the RQ-formulation of several QR-based algorithms, and an recursive algorithm to implement Step 2. We will then present a new algorithm based on the RQ-formulation of the recursive algorithm, thus presenting a link between these apparently different algorithms.

Third. In the section 5, we will prove that if Step 2 is implemented in a numerically stable way, then the overall algorithm will be numerically stable, thus reproving the numerical stability of several known QR-based algorithms and proving that of the new algorithm.

2.1 The Method of Miminis and Paige (Miminis and Paige (1981))

The basic idea of the method is to apply the QR Algorithm with ultimate shifts to the matrix (with unknown first row) $(H - \beta e_1 k^t)$. If for simplicity we assume that the closed-loop eigenvalues are all real, then the method consists of n deflation steps and a "backward sweep". Each deflation step can be thought of as an RQ factorization of the matrix $(H_i - \beta_i e_1 k_i^t - \lambda_i I)$. However, since k_i is unknown, the process is not quite so straightforward. We first compute Q_i such that $(H_i - \lambda_i I)Q_i^t = R_i$ is upper triangular. Then $U_i = (H_i - \beta_i e_1 k_i^t - \lambda_i I)Q_i^t$ must also be upper triangular, and we want to choose k_i such that U_i is singular. Now since H_i is unreduced, the only way that U_i can be singular is if $U_i e_1 = 0$, that is, if $u_{11}^{(i)} \equiv e_1^t U_i e_1 = 0$. Write $Q_i = \begin{bmatrix} y_i^t \\ \tilde{Q}_i \end{bmatrix}$. Then

$$0 = U_i e_1 = (H_i - \beta_i e_1 k_i^t - \lambda_i I) y_i,$$

or

$$\beta_i k_i^t y_i = e_1^t (H_i - \lambda_i I) y_i = r_{11}^{(i)}. \tag{2.1}$$

This is a key relation in the method, but it does *not* allow us to compute k_i, so we continue. To complete the RQ-step we premultiply by Q_i and add back the λ_i to get

$$Q_i (H_i - \beta_i e_1 k_i^t) Q_i^t = \begin{bmatrix} \lambda_i & * \\ 0 & \tilde{Q}_i (H_i - \beta_i e_1 k_i^t) \tilde{Q}_i^t \end{bmatrix}.$$

Now if we define $H_{i+1} = \tilde{Q}_i H_i \tilde{Q}_i^t, \beta_{i+1} = q_{21}^{(i)} \beta_i$, and $k_{i+1} = \tilde{Q}_i k_i$, then the ith deflation step is complete; H_{i+1} is unreduced, β_{i+1} is non-zero, and we can continue with the controllable pair $(H_{i+1}, \beta_{i+1} e_1)$ and the *unknown* feedback vector k_{i+1} of dimension one less than that of k_i.

At the final deflation step we have $H_n - \beta_n e_1 k_n^t \in \boldsymbol{R}^{1\times 1}$, i.e, $k_n = (H_n - \lambda_n I)/\beta_n$ is a real number.

The backward sweep consists of computing $k_{n-1}, k_{n-2}, \ldots, k_1 = k$ using the relations

$$k_{i+1} = \tilde{Q}_i k_i \tag{2.2}$$

and from (2.1)

$$y_i^t k_i = r_{11}^{(i)} / \beta_i. \tag{2.3}$$

Combining these equations we have

$$k_i = Q_i^t \begin{pmatrix} r_{11}^{(i)} / \beta_i \\ k_{i+1} \end{pmatrix}, i = n-1, n-2, \ldots, 1. \tag{2.4}$$

We summarize the preceding discussion as an algorithm:

Algorithm 2.2 The RQ Formulation of Single-input Algorithm of Miminis and Paige

Input: H, an unreduced $n \times n$ Hessenberg matrix, $\beta \neq 0$,
and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: k such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 1 Set $H_1 = H$, and $\beta_1 = \beta$
For $i = 1, 2, \ldots, n-1$ do
Compute $(H_i - \lambda_i I)Q_i^t = R_i$, the RQ factorization of $(H_i - \lambda_i I)$
Compute $\tau_i = r_{11}^{(i)}/\beta_i$ and $\beta_{i+1} = q_{21}^{(i)}\beta_i$
Compute H_{i+1}, where $Q_i R_i + \lambda_i I = \begin{bmatrix} * & * \\ 0 & H_{i+1} \end{bmatrix}$
End

Step 2 Compute $k_n = (H_n - \lambda_n)/\beta_n$
For $i = n-1, n-2, \ldots, 1$ do
Compute $k_i = Q_i^t \begin{pmatrix} \tau_i \\ k_{i+1} \end{pmatrix}$
End

Flop-count: When implemented with implicit double steps, this algorithm takes about $\frac{5}{6}n^3$ flops. Combined with the $\frac{5}{3}n^3$ flops required for the controller-Hessenberg reduction, the total flop count is about $\frac{5}{2}n^3$.

2.2 The Method of Petkov, Christov and Konstantinov (Petkov, Christov and Konstantinov (1984)).

This method, like the Miminis-Paige method, is based on an ultimately shifted RQ step with immediate deflation. The only real difference between the two methods is *how* the matrices Q_i are computed. In fact, we will show by the end of this section that the Q_i obtained by these methods are essentially the same throughout the entire deflation sequence. We will devote a major portion of this section to an analysis of the RQ factorization (and therefore the deflation step) in the method of Petkov, Christov and Konstantinov.

If λ is an eigenvalue of the Hessenberg matrix $(H - \beta e_1 k^t)$, then there exists $v \neq 0$ such that

$$(H - \lambda I)v = \beta e_1 k^t v. \tag{2.5}$$

Now partition $(H-\lambda I)$ and v as

$$(H-\lambda I) = \begin{bmatrix} * & * \\ T & c \end{bmatrix} \textit{ and } v = \begin{bmatrix} \tilde{v} \\ v_k \end{bmatrix},$$

where $T \in \boldsymbol{R}^{n-1\times n-1}$ is upper triangular. Then from (2.5) we have $[T\ c]v = 0$, or

$$T\tilde{v} = -v_k c.$$

Since H is unreduced, T is nonsingular and v_k is nonzero; so if we fix $v_k \neq 0$, we can compute v by back-substitution. We now have an eigenvalue/eigenvector pair (λ, v) of the matrix $(H - \beta e_1 k^t)$, and if we can compute an orthogonal matrix Q such that $Qv = \alpha e_1$ and $Q(H-\lambda I)Q^t$ is a Hessenberg matrix, then

$$0 = (H - \beta e_1 k^t - \lambda I)v = (H - \beta e_1 k^t - \lambda I)\alpha Q^t e_1,$$

or

$$(H-\lambda I)Q^t e_1 = \beta(k^t Q^t e_1)e_1. \tag{2.6}$$

If we now write $Q = \begin{bmatrix} y^t \\ \tilde{Q} \end{bmatrix}$, then (2.6) yields

$$\beta k^t y = e_1^t (H-\lambda I)y.$$

Inserting subscripts and continuing in the fashion of the last section, we see that

$$Q_i(H_i - \beta_i e_1 k_i^t)Q_i^t \quad = \quad \begin{bmatrix} \lambda_i & * \\ 0 & \tilde{Q}_i(H_i - \beta_i e_1 k_i^t)\tilde{Q}_i^t \end{bmatrix}.$$

Define $H_{i+1} = \tilde{Q}_i H_i \tilde{Q}_i^t, \beta_{i+1} = q_{21}^{(i)}\beta_i$, and $k_{i+1} = \tilde{Q}_i k_i$. If Q_i is unreduced, then H_{i+1} is also unreduced, β_i is nonzero, and therefore the pair $(H_{i+1}, \beta_{i+1}e_1)$ is controllable. This is entirely the same situation as in the method of Miminis and Paige, and as such we can use the same backward sweep to recover $k = k_1$.

We have not yet explained how to compute an orthogonal matrix Q such that $Qv = \alpha e_1$ and $Q(H-\lambda I)Q^t$ is a Hessenberg matrix; the following lemma illustrates the construction.

Lemma 2.1 *Let $Hv = \lambda v$, where H is an unreduced upper Hessenberg matrix. Let the Givens rotations J_k in the k and $(k+1)^{st}$ planes be such that $J_i J_{i+1} \cdots J_{n-1} v = (x_i^t, \alpha_i, 0)^t$ for $i = n-1, n-2, \ldots, 1$, where $x_i \in \boldsymbol{R}^{i-1\times 1}$ and $\alpha_i \in \boldsymbol{R}$. Then*

$$(H-\lambda I)J_{n-1}^t J_{n-2}^t \cdots J_1^t = R \tag{2.7}$$

is upper triangular.

Proof: Define $M_i = (H - \lambda I)J_{n-1}^t J_{n-2}^t \cdots J_i^t$ and suppose that

$$M_i = \begin{bmatrix} k_i & * & * \\ 0 & y_i^t & * \\ 0 & 0 & R_i \end{bmatrix},$$

where $y_i \in R^{2\times 1}, R_i \in R^{n-i\times n-i}$ is upper triangular, and $\begin{bmatrix} k_i & * \\ 0 & y_i^t \end{bmatrix}$ is an unreduced upper Hessenberg matrix of order i. We will show that $\begin{bmatrix} k_{i-1} & * \\ 0 & y_{i-1}^t \end{bmatrix}$ is also an unreduced upper Hessenberg matrix and that R_{i-1} is upper triangular. Thus, by induction we will have (2.7). Now

$$M_i J_{i-1}^t = \begin{bmatrix} k_i & * & * \\ 0 & \bar{y}_i^t & * \\ 0 & 0 & R_i \end{bmatrix},$$

and since $(H - \lambda I)v = 0$ we must have that

$$0 = M_{i-1}(J_{i-1}J_i \cdots J_{n-1}v) = M_{i-1}\begin{pmatrix} x_{i-1} \\ \alpha_{i-1} \\ 0 \end{pmatrix}.$$

Therefore $\bar{y}_i^t$ must be of the form $\bar{y}_i^t = (0, y) \in R^{1\times 2}$, with

$$M_{i-1} = \begin{bmatrix} k_{i-1} & * & * \\ 0 & y_{i-1}^t & * \\ 0 & 0 & R_{i-1} \end{bmatrix},$$

R_{i-1} upper triangular, and

$$\begin{bmatrix} k_{i-1} & * \\ 0 & y_{i-1}^t \end{bmatrix}$$

unreduced upper Hessenberg. This completes the induction step; and since M_1 is an unreduced upper Hessenberg matrix, the proof is complete.

Quite simply, the rotations J_i that are defined by v provide the RQ factorization: $(H - \lambda I)J_{n-1}^t J_{n-2}^t \cdots J_1^t = (H - \lambda I)Q^t = R$.

We summarize the Petkov-Christov-Konstantinov method:

Algorithm 2.3 The RQ-Formulation of the Single Input Method of Petkov, Christov and Konstantinov

Input: H, an unreduced $n \times n$ Hessenberg matrix, $\beta \neq 0$,
and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: k such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 1 Set $H_1 = H$, and $\beta_1 = \beta$

For $i = 1, 2, \ldots, n-1$ do

Compute $(H_i - \lambda_i I)Q_i^t = U_i$, the RQ factorization of $(H_i - \lambda_i I)$ by computing v_i such that $(H_i - \lambda_i I)v_i = \gamma e_1$ and then computing the rotations J_i such that $J_1 J_2 \cdots J_{n-i+1} v_i = \pm ||v_i||_2 e_1$, and finally setting $Q_i = J_1 J_2 \cdots J_{n-i+1}$

Compute $\tau_i = e_1^t (H_i - \lambda_i I) Q_i^t e_1 / \beta_i$ and $\beta_{i+1} = q_{21}^{(i)} \beta_i$

Compute H_{i+1}, where $Q_i U_i + \lambda_i I = \begin{bmatrix} * & * \\ 0 & H_{i+1} \end{bmatrix}$

End

Step 2 Compute $k_n = (H_n - \lambda_n)/\beta_n$

For $i = n-1, n-2, \ldots, 1$ do

Compute $k_i = Q_i^t \begin{pmatrix} \tau_i \\ k_{i+1} \end{pmatrix}$

End

Flop-count: If implemented with care, this algorithm takes about $\frac{5}{3}n^3$ flops. When combined with the $\frac{5}{3}n^3$ flops required for the controller-Hessenberg reduction, the total flop count is about $\frac{10}{3}n^3$.

2.3 The Method of Patel and Misra (Patel and Misra (1984)).

We have now seen two methods based on an explicit RQ step with immediate deflation. It should come as no surprise that an implicit RQ step is possible, and in order to handle complex pairs of eigenvalues with real arithmetic, an implicit double step is needed. Such a method was proposed by (Patel and Misra (1984)). The method is similar to the method of Miminis and

Paige (1982), but it includes an alternative to the "backward sweep", and is the first published description of the implicit double-step in the single-input eigenvalue assignment problem. We will outline an implicit single-step here.

First, compute an orthogonal matrix P_i such that $e_n^t(H_i - \lambda_i I)P_i^t = \alpha e_n^t$; then compute another orthogonal matrix U_i such that $U_i P_i H_i P_i^t U_i^t$ is an upper Hessenberg matrix; finally, set $Q_i = U_i P_i$. The matrix U_i "chases the bulge" *up* the subdiagonal of $P_i H_i P_i^t$. We are now in the familiar situation of computing k_i such that

$$Q_i(H_i - \beta_i e_1 k_i^t)Q_i^t = \begin{bmatrix} \lambda_i & * \\ 0 & H_{i+1} - \beta_{i+1} e_1 k_{i+1}^t \end{bmatrix}.$$

If, as before, we set $Q_i = \begin{bmatrix} y_i^t \\ \tilde{Q}_i \end{bmatrix}$, then with $\tau_i \equiv k_i^t y_i$, we must have

$$\tau_i = \frac{\lambda_i - h_{11}^{(i)}}{\beta_i q_{11}^{(i)}} = \frac{h_{21}^{(i)}}{\beta_i q_{21}^{(i)}},$$

with the continuation $H_{i+1} = \tilde{Q}_i H_i \tilde{Q}_i^t, \beta_{i+1} = q_{21}^{(i)}\beta_i$, and $k_{i+1} = \tilde{Q}_i k_i$. After n such steps we expect the usual backward sweep, but it is shown in Patel and Misra (1984) that this computation need not be put off that long: the backward sweep

$$k_i = Q_i^t \begin{pmatrix} \tau_i \\ k_{i+1} \end{pmatrix}, i = n-1, n-2, \ldots, 1$$

is equivalent to the "forward update"

$$\hat{Q} = I,\ k = 0 \ and$$

$$k^t = k^t + \tau_i y_i \hat{Q},\ \hat{Q} = \tilde{Q}_i \hat{Q},\ i = 1, 2, \ldots, n-1.$$

Algorithm 2.4 The Single Input Algorithm of Patel and Misra

Input: H, an unreduced $n \times n$ Hessenberg matrix, $\beta \neq 0$,
and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: k such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 1 Set $H_1 = H, \beta_1 = \beta, \hat{Q} = I$, and $k = 0$

For $i = 1, 2, \ldots, n-1$ do

Compute P_i such that $e_n^t(H_i - \lambda_i I)P_i^t = \alpha e_n^t$

Compute U_i such that $U_i P_i H_i P_i^t U_i^t$

is an upper Hessenberg matrix

Set $Q_i = \begin{bmatrix} y_i^t \\ \tilde{Q}_i \end{bmatrix} = U_i P_i$

Compute $\tau = e_1^t(H_i - \lambda_i I)Q_i^t e_1/\beta_i$

Compute $\beta_{i+1} = q_{21}^{(i)}\beta_i$

Compute H_{i+1}, where $Q_i H_i Q_i^t = \begin{bmatrix} * & * \\ 0 & H_{i+1} \end{bmatrix}$

Compute $k = k + \tau\hat{Q}^t y_i^t$

Compute $\hat{Q} = \tilde{Q}_i\hat{Q}$

End

Step 2 Compute $\tau = (H_n - \lambda_n)/\beta_n$

Compute $k = k + \tau\hat{Q}^t$

Flop-Count: If implemented with implicit double steps, this algorithm is the same as the method of Miminis and Paige (1982), except for the forward update/backward sweep, which either way is $O(n^2)$ flops. Therefore, this method requires about $\frac{5}{6}n^3$ flops. When combined with the $\frac{5}{3}n^3$ flops required for the controller- Hessenberg reduction, the total flop count is about $\frac{5}{2}n^3$.

2.4 A Recursive Algorithm (Datta (1987))

We reproduce below the recursive algorithm of Datta (1987), which is apparently different from the three just described, and show how this algorithm produces an explicit formula for the single-input feedback vector.

In the next section, we will present an RQ-formulation of this method. This new RQ method will help elucidate the relationship between the other RQ methods and the explicit expression for the feedback vector obtained by the recursive formula.

Algorithm 2.5 A Recursive Algorithm (Datta (1987))

Input: H, an unreduced $n \times n$ Hessenberg matrix, $\beta \neq 0$, and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: k such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 1 Set $l_1 = e_n$

Step 2 For $i = 1, 2, \ldots, n-1$ do

Compute $\hat{l}_{i+1} = (H^t - \lambda_i I) l_i$

Compute $l_{i+1} = b_i \hat{l}_{i+1}$, where b_i is chosen so that $||l_i|| \in (\frac{1}{2}, 1]$, say

End

Step 3 Compute $k = \frac{1}{\beta l_{1n}} (H^t - \lambda_n I) l_n$

Flop-count: This method requires only about $\frac{1}{6}n^3$ flops. When combined with the $\frac{5}{3}n^3$ flops required for the controller-Hessenberg reduction, the total flop count is about $\frac{11}{6}n^3$. Given a system in controller-Hessenberg form, this method is more than five times as fast as the method of Miminis and Paige. The assignment of complex pairs of eigenvalues in real arithmetic requires a slight adjustment to the above method, but does not alter the operations count.

A Closed-Form Solution of the Single-input EAP

We now show that this method yields an explicit closed-form solution for the single-input problem. The recursion in step 2 above yields

$$\gamma l_{i+1} = (H^t - \lambda_1 I)(H^t - \lambda_2 I) \cdots (H^t - \lambda_i I) l_1, \tag{2.8}$$

and including steps 1 and 3, (2.8) becomes

$$\alpha k = (H^t - \lambda_1 I)(H^t - \lambda_2 I) \cdots (H^t - \lambda_n I) e_n, \tag{2.9}$$

where $\alpha = (\beta h_{21} h_{32} \cdots h_{n,n-1})^{-1}$. If $\phi(x) = (x - \lambda_1)(x - \lambda_2) \cdots (x - \lambda_n)$, then this will be written as

$$k^t = \alpha e_n^t \phi(H). \tag{2.10}$$

Since this solution is unique, it represents *the* Hessenberg formula for the single-input EAP.

2.5 Of Methods not Discussed

A. Varga (1981) proposed a method very different from those considered here. It has largely been ignored by numerical linear algebraists because of a reduction of the original system to controller-Schur form ($T = PAP^t$ is block upper triangular with 1×1 or 2×2 diagonal blocks, and $k = Pb$ is a "full" column vector, see Varga (1981) for details). It is argued that, besides the extra work involved, the method suffers from the fact that possible illconditioning of the eigenvalues of the original system introduces unnecessary errors into the computation. These criticisms, while entirely valid from an algorithmic perspective, may be unwarranted from a more global view. It may be that knowledge of the original spectra (provided by the controller-Schur form and *not* by the controller-Hessenberg form) is necessary for intelligent Eigenvalue Assignment. In that case the information provided by the Schur decomposition might be used in choosing Ω. If the eigenvalues of the original system were found to be illconditioned, a Hessenberg method might be preferable; but if not, continuing on with the method of Varga would be more efficient.

There exist many methods for the Eigenvalue Assignment problem, and we have chosen to discuss only those few with positive numerical attributes (e.g. stability and efficiency). Methods that depend on Jordan or Frobenius forms are both expensive and unstable. Most closed-form solutions for the feedback vector require such forms and hence lead to poor numerical methods. One of the most well known closed-form solutions is due to Ackermann (1972); while it is often held as an example of how *not* to solve the EAP, we will see in the next section that each of the methods discussed in this paper are closely related to that solution.

3 Relationship Between the Various Methods

In this section we will explain the relationships between the methods of Miminis and Paige; Petkov, Christov, and Konstantinov; Patel and Misra; and Datta. We will show that the Miminis-Paige, Petkov-Christov-Konstantinov, and Patel-Misra methods yield the same data at each deflation step, the only difference being the technique used for an RQ factorization. Then we will present an RQ implementation of the recursive method that ties all four of the methods together.

The Miminis-Paige, Petkov-Christov-Konstantinov, and Patel-Misra methods all have an RQ factorization at the heart of the deflation step. With the original method of Miminis and Paige we have the explicit Hessenberg RQ factorization, with that of Petkov, Christov and Konstantinov we have a novel "triangular system" Hessenberg RQ factorization, and with the method of Patel and Misra we have the implicit Hessenberg RQ factorization.

These methods all begin with the same data, the pair $(H, \beta e_1)$ and the closed-loop spectrum Ω; furthermore it is clear that each of the methods generates the $i+1^{st}$ set of data by applying an RQ iteration step to the i^{th} set of data. Thus, given the matrix H_i, the Implicit-Q Theorem (or the uniqueness of the RQ factorization) guarantees that whichever method we choose, the unreduced Hessenberg matrix H_{i+1} is essentially (that is, up to a diagonal scaling of ± 1) the same. One might question the uniqueness of the RQ factorization (or equivalently, the implicit RQ step) if λ_i is an eigenvalue of H_i. Indeed, in this case it is not unique, for while Q_i *is completely determined*, the first row of R_i is underdetermined. But if we now note that the deflation step is taken immediately in each of the methods, it is clear that the first row of R_i *plays no part in the computation.* We have proven the following lemma:

Lemma 3.1 *In exact arithmetic, the methods of Miminis and Paige, Petkov, Christov and Konstantinov and Patel and Misra all generate the same data H_i, and Q_i at each deflation step, up to a sign scaling, for $i = 1, 2, \ldots, n$.*

The differences between these methods, *at each* step, depend only on finite precision. The discussion above allows us to give a generic formulation of all of the QR-based single-input algorithms as follows:

Algorithm 3.1 Generic RQ-based Single Input Algorithm

Input: H, an unreduced $n \times n$ Hessenberg matrix, $\beta \neq 0$,
and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: k such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 1 Set $H_1 = H$ and $\beta_1 = \beta$

For $i = 1, 2, \ldots, n-1$ do

Compute Q_i from a shifted RQ step with H_i and λ_i:

$\tilde{H}_i = Q_i H_i Q_i^t$

Compute $\tau_i = \frac{\tilde{h}_{21}^{(i)}}{\beta_i q_{21}^{(i)}} = \frac{\lambda_i - \tilde{h}_{11}^{(i)}}{\beta_i q_{11}^{(i)}}$

Compute $\beta_{i+1} = \beta_i q_{21}^{(i)}$

Compute H_{i+1}, where $\tilde{H}_i = \begin{bmatrix} * & * \\ 0 & H_{i+1} \end{bmatrix}$

End

Step 2 Compute $\tau_n = \frac{\lambda_n - H_n}{\beta_n}$

Step 3 Compute $k^t = (\tau_1, \tau_2, \ldots, \tau_n) Q_{n-1} Q_{n-2} \cdots Q_1$

The manner of computing the RQ steps in this generic method is not specified; an explicitly shifted RQ step with Givens rotations yields the original method of Miminis-Paige, explicitly computing the RQ factors using a closed-loop eigenvector gives the method of Petkov-Christov-Konstantinov, and an implicit RQ step corresponds to the methods of Patel-Misra and Miminis-Paige. We also note that τ_i can be computed using either of the quantities given above, or if R_i is available, as $\tau_i = r_{11}^{(i)}/\beta_i$.

4 A New RQ-based Method

We now present a new RQ implementation of the recursive algorithm of Datta that will make explicit the connections between all of these methods and the explicit formula (2.10).

While this method was discovered and proved in the context of the matrix equation

$$H^t L - LB = ce_n^t,$$

we can show its relationship with the often maligned formula of Ackermann. Ackermann (1972) showed that if

$$\phi(x) = (x - \lambda_1)(x - \lambda_2) \cdots (x - \lambda_n),$$

then the unique solution to the EAP for the controllable pair (A, b) is

$$f^t = e_n^t C^{-1} \phi(A), \tag{4.1}$$

where

$$C \equiv \begin{bmatrix} b, & Ab, & \cdots & A^{n-1}b \end{bmatrix}.$$

If $(H, \beta e_1) = (PAP^t, Pb)$ is the controller-Hessenberg form of (A, b), then from (4.1)

$$\begin{aligned} f^t P^t &= e_n^t C^{-1} \phi(A) P^t \\ &= e_n^t C^{-1} P^t \phi(H) \\ &= e_n^t (PC)^{-1} \phi(H), \end{aligned}$$

where PC is an upper triangular matrix. If α^{-1} is the (n, n) element of PC, then $e_n^t (PC)^{-1} = \alpha e_n^t$, and we see that the formula, $k^t = \alpha e_n^t \phi(H)$ is a Hessenberg case of Ackermann's formula (in fact $\alpha^{-1} = \beta \prod_{i=1}^{n-1} h_{i+1,i}$).

The recursive algorithm is an extremely efficient way to solve the Hessenberg Single Input problem, but as we will see in Chapter 6, backward stability cannot be guaranteed. Having been aware of possible instabilities in the recursive formulation, Datta (1992) suggested that this method could be implemented using QR iterations as follows:

Set $H_1 = H$

For $i = 1, 2, \ldots, n$ compute the QR step

$$\begin{aligned} Q_i R_i &:= H_i - \lambda_i I \\ H_{i+1} &:= R_i Q_i + \lambda_i I \end{aligned}$$

Then it can be shown (Stewart [1972, 353]) that

$$\phi(H) = Q_1 Q_2 \cdots Q_n R_n R_{n-1} \cdots R_1,$$

and setting $Q = Q_1 Q_2 \cdots Q_n$ and $R = R_n R_{n-1} \cdots R_1$, formula (2.10) becomes

$$k^t = \alpha e_n^t Q R.$$

The difficulty of implementing this strategy is that the R_i need to be accumulated; this is both expensive and unstable.

We now show how the method can be made computationally efficient by using RQ factorizations instead of the QR factorizations.

Set $H_1 = H$

For $i = 1, 2, \ldots, n$ compute the RQ step

$$\begin{aligned} R_i Q_i &:= H_i - \lambda_i I \\ H_{i+1} &:= Q_i R_i + \lambda_i I \end{aligned}$$

This time

$$\phi(H) = R_1 R_2 \cdots R_n Q_n Q_{n-1} \cdots Q_1, \tag{4.2}$$

and by setting $Q = Q_n Q_{n-1} \cdots Q_1$ and $R = R_1 R_2 \cdots R_n$, we have

$$k^t = \alpha e_n^t R Q \quad = \quad \alpha \rho e_n^t Q,$$

where $\rho = \prod_{i=1}^n r_{nn}^{(i)}$. This is a much nicer situation! Furthermore, we will now show that it is possible to "deflate" the problem at each RQ step.

Write $Q_i, i = 1, 2, \ldots, n$ as a product of Givens rotations $Q_i = J_1^{(i)} J_2^{(i)} \cdots J_{n-1}^{(i)}$, where $J_k^{(i)}$ is a rotation in the k and $k+1$ planes. Then

$$\begin{aligned} e_n^t Q &= e_n^t J_1^{(n)} J_2^{(n)} \cdots J_{n-1}^{(n)} Q_{n-1} Q_{n-2} \cdots Q_1 \\ &= e_n^t J_{n-1}^{(n)} Q_{n-1} Q_{n-2} \cdots Q_1 \\ &= e_n^t J_{n-1}^{(n)} J_{n-2}^{(n-1)} J_{n-1}^{(n-1)} Q_{n-2} Q_{n-3} \cdots Q_1 \\ &\vdots \\ &= e_n^t (J_{n-1}^{(n)})(J_{n-2}^{(n-1)} J_{n-1}^{(n-1)}) \cdots (J_1^{(2)} J_2^{(2)} \cdots J_{n-1}^{(2)}) Q_1, \end{aligned} \tag{4.3}$$

or $k^t = \alpha\rho e_n^t \hat{Q}_n \hat{Q}_{n-1} \cdots \hat{Q}_1$, where $\hat{Q}_k \equiv J_{k-1}^{(k)} J_k^{(k)} \cdots J_{n-1}^{(k)}$. Algorithmically we have

Algorithm 4.1 A Proposed Single Input Algorithm

Input: H, an unreduced $n \times n$ Hessenberg matrix, $\beta \neq 0$,
and $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$

Output: k such that $\lambda(H - \beta e_1 k^t) = \Omega$

Step 1 Compute $R_1 Q_1 := H - \lambda_1 I$, the RQ factorization of $H - \lambda_1 I$
Compute $H_2 = Q_1 R_1 + \lambda_1 I$
Set $Q = Q_1$ and $\rho = r_{nn}^{(1)}$

Step 2 For $i = 2, 3, \ldots, n-1$
Compute $R_i Q_i := H - \lambda_i I$
Compute H_{i+1},where $Q_i R_i + \lambda_i I = \begin{bmatrix} * & * \\ 0 & H_{i+1} \end{bmatrix}$
Update $Q := \begin{bmatrix} I & \\ & Q_k \end{bmatrix} Q$
Update $\rho := \rho r_{n+2-i,n+2-i}^{(i)}$ ($r_{n+2-i,n+2-i}^{(i)}$ is the last element of R_i)
End

Step 3 Update $\rho := \rho(H_n - \lambda_n)$
Compute $k^t = \alpha\rho e_n^t Q$

Remark: The RQ factorizations in this method can be implemented implicitly with a double step, but as with the previous methods this has been omitted for the sake of clarity. Note also that there are no divisions in this method until α is computed in the last step, i.e. the troublesome computation of

$$\tau_i = \frac{\tilde{h}_{21}^{(i)}}{\beta_i q_{21}^{(i)}} = \frac{\lambda_i - \tilde{h}_{11}^{(i)}}{\beta_i q_{11}^{(i)}} = \frac{r_{11}^{(i)}}{\beta_i}$$

which appeared in the other methods does not appear here.

Flop-count: Implemented using implicit double-steps with Q kept in factored form, this method requires about $\frac{5}{6}n^3$ flops. When combined with the $\frac{5}{3}n^3$ flops required for the controller-Hessenberg reduction, the total flop count is about $\frac{5}{2}n^3$, the same as the the Miminis-Paige and Patel-Misra methods.

4.1 A Relationship Between the Proposed Method and other RQ Methods

We promised that this method would shed some light on the relationship between the other RQ methods and the closed-form solution (2.10). While the connection between this RQ method and the closed-form solution is clear, we have yet to close the final link. There are two differences between this method and the generic RQ method: (i) deflation does not commence after the first iteration here as in the generic method, and (ii) the scalar $\alpha\rho$ in this method takes the place of the vector $x^t = (\tau_1, \tau_2, \ldots, \tau_n)$.

When viewed from the perspective of the generic method, these two distinctions are the result of transforming the vector x^t into the vector $\alpha\rho e_n^t$, one step at a time.

To see how this works let us consider an explicit RQ factorization of H_i, an unreduced Hessenberg matrix of order k, say. In the generic method the RQ factorization effectively stops when the matrix $H_i Q_i^t$ is of the form

$$H_i Q_i^t = \begin{bmatrix} A & * \\ 0 & T \end{bmatrix}, \tag{4.4}$$

where A is 2×2 and T is upper triangular. In the proposed RQ method above, we are one deflation step behind, so that one more rotation is needed to put A into triangular form. This rotation V_i will be such that $H_i Q_i^t V_i^t$ is upper triangular; but it also rolls $[\tau_i, \tau_{i+1}]$ into $[0, \gamma]$, for

$$\begin{aligned} [a_{21}, a_{22}] &= [r_{11}^{(i)} \tfrac{\beta_{i+1}}{\beta_i}, r_{11}^{(i+1)}] \\ &= \beta_{i+1}[\tfrac{r_{11}^{(i)}}{\beta_i}, \tfrac{r_{11}^{(i+1)}}{\beta_{i+1}}] \\ &= \beta_{i+1}[\tau_i, \tau_{i+1}]. \end{aligned} \tag{4.5}$$

We have proven the following theorem:

Theorem 4.1 *The generic RQ method generates the orthogonal matrices Q_i that satisfy*

$$k^t = (\tau_1, \tau_2, \ldots, \tau_n) Q_n Q_{n-1} \cdots Q_1,$$

while the proposed method generates the orthogonal matrices $P_i = Q_i V_i$ such that

$$k^t = \alpha\rho e_n^t P_n P_{n-1} \cdots P_1,$$

where

$$(\tau_1, \tau_2, \ldots, \tau_n) V_1 V_2 \cdots V_{n-1} = \alpha\rho e_n^t$$

and V_i is a rotation in the planes i and $i+1$.

5 Error Analysis

A systematic round-off error analysis of most of the existing and currently used algorithms in control theory is lacking. As far as algorithms for the EAP is concerned, round-off error analyses of only the methods of Miminis and Paige and Petkov, Christov and Konstantinov have been presented (Cox and Moss (1989 and 1992), Miminis and Paige (1988)).

In this section we give a detailed round-off error analysis of our proposed single-input algorithm (Algorithm 4.1) described in Section 4, and prove that it is backward stable. In the course of this proof we show that any algorithm for the EAP is backward stable if it is backward stable for the corresponding Hessenberg problem. We then give a round-off error analysis of the recursive algorithm. Our analysis shows that the latter, while it may not be backward stable, is reliable in the sense that we can detect precisely when the results are suspect.

5.1 An Error Analysis of the Proposed Single Input Method

The backward error analysis of eigenvalue assignment methods has turned out to be a non-trivial task. For example, the RQ-based methods of Section 2 are straightforward adaptations of the Hessenberg QR iteration, and while a backward error analysis for the QR iteration is quite simple, that for the eigenvalue assignment methods is not (see e.g. Cox and Moss (1989 or 1992)). The major difference is that backward error analysis for the QR iteration in the eigenvalue problem is naturally focused on showing that the next iterate is (exactly) similar to a matrix that is close to the current iterate, while for eigenvalue assignment we cannot, in a straightforward way, use similarity as a tool. In order to simplify the analysis, we show that backward stability is achieved if one can solve the Hessenberg single-input eigenvalue assignment problem in a backward stable manner. First we prove that Algorithm 4.1 is backward stable.

Theorem 5.1 *The RQ-based single-input eigenvalue assignment (Algorithm 4.1) is backward stable, i.e. it computes a feedback k such that*

$$\lambda(H + \delta H - (\beta + \delta\beta) e_1 (k + \delta k)^t) = \Omega,$$

where δH, $\delta\beta$, and δk are small.

Proof: Let $\bar{H}_1 = H_1 = H$, and let $\bar{H}_i$ be the computed iterate at the i^{th} QR step. Let $\bar{Q}_i$ be the computed transformation at each step. Then we have from basic error analysis (see e.g. [Wilkinson (1965), 110-160]) that there exists an orthogonal matrix Q_1 such that

$$\bar{H}_2 = Q_1 (H_1 + \delta H_1) Q_1^t, \quad \bar{Q}_1 + E_1 = Q_1,$$

where $||\delta H_1||_F \leq f(n)\boldsymbol{u}max\{||H_1||_F, |\lambda_i|\}$, and $||E_1||_F \leq g(n)\boldsymbol{u}$, where g and f are modest functions of n, practically behaving like cn, with c a constant of order unity. Note that with an implicit double RQ step, the upper bound on $||\delta H_1||_F$ is independent of λ_i. Now iterating on these results leads to

$$\bar{H}_n = P_{n-1}(H_1 + \delta H)P_{n-1}^t, \quad \bar{P}_{n-1} + E = P_{n-1}, \tag{5.1}$$

where $P_{n-1} = Q_{n-1}Q_{n-2}\cdots Q_1$, $\bar{P}_{n-1} = fl(\bar{Q}_{n-1}\bar{Q}_{n-2}\cdots\bar{Q}_1)$. Here, $||\delta H||_F \leq \boldsymbol{u}nf(n)$ $max\{||H||_F, |\lambda_k|, k = 1, 2, \ldots, n\}$ and $||E||_F \leq ng(n)\boldsymbol{u}$. These bounds are pessimistic now, because of the maximum over $|\lambda_k|$ and because we have not considered the fact that H_i and Q_i are actually $(n-i+1)\times(n-i+1)$, not $n \times n$. Now with $P = P_n$, the feedback k is computed as $k^t = \gamma e_n^t P$, so up to the scalar γ, we are done. We have shown that $e_n^t(\bar{P} + E)$ is exact for a matrix $H + \delta H$, where $||e_n^t E|| \leq ng(n)\boldsymbol{u}$ and $||\delta H|| \leq \boldsymbol{u}nf(n)$ $max\{||H||_F, |\lambda_k|\}$. We now show that γ can be computed in a backward stable fashion, thereby completing the proof.

Remember that $\gamma = \alpha\rho$, where $\rho = \prod_{i=1}^n r_i$, $\alpha = (\prod_{i=1}^n \beta_i)^{-1}$, $\beta_i = h_{i,i-1}$, $\beta_0 = \beta$, and $r_i = r_{nn}^{(i)}$ is the (n,n) entry of R_i. Now let $\bar{r}_i$ be the computed value of r_i, where R_i is exact for the computed matrix $\bar{H}_i$. Then the errors *not accounted for* in $||\delta H||$ can be expressed as

$$\bar{r}_i = \pm\sqrt{h_{n,n-1}^2(1+\epsilon_1) + [(h_{nn} - \lambda_i)(1+\epsilon_2)]^2(1+\epsilon_3)}(1+\epsilon_4),$$

where $|\epsilon_j| < \boldsymbol{u}$. Write this as $\bar{r}_i = r_i(1+\delta_i)$, where $|\delta| \leq \frac{5}{2}\boldsymbol{u}$. Now $\bar{\gamma} = fl(\bar{\rho}\bar{\alpha})$ so that

$$\bar{\gamma} = \prod_{i=1}^n \frac{r_i(1+\delta_i)(1+\tau_i)}{\beta_i(1+\epsilon_i)},$$

where $|\tau_i|, |\epsilon_i| \leq \boldsymbol{u}$, $i = 1, 2, \ldots, n$. Therefore,

$$\gamma - \bar{\gamma} = \gamma \prod_{i=1}^n \frac{(1+\delta_i)(1+\tau_i)}{(1+\epsilon_i)},$$

and if we assume that $n\boldsymbol{u} < 0.1$, then conservatively

$$|\gamma - \bar{\gamma}| \leq 5n\boldsymbol{u}|\gamma|.$$

If we write $\gamma = \gamma(\beta, H)$, then our result reads

$$\bar{\gamma} = \gamma(\beta + \hat{\delta\beta}, H + \delta H),$$

where δH is the same as in (5.1), and $|\hat{\delta\beta}| \leq 5n\boldsymbol{u}$. Finally, the error from the scalar-vector operation $k^t = \gamma e_n^t P$ can be thrown back into β, yielding a computed feedback $\bar{k}$ such that

$$\bar{k} = \bar{\gamma}e_n^t E + k(\beta + \delta\beta, H + \delta H),$$

with $|\delta\beta| \leq 5n\boldsymbol{u}$.

Remark: The popular definition for backward stability is not used here for a very simple reason. Consider proving that the computation of a Householder reflection is backward stable. One must show that the computed matrix is exact for a problem close to the original. This is impossible, even for the $n = 2$ case, for the computed matrix is almost always *not an orthogonal matrix.* This difficulty is removed by adopting the more general definition of Stewart [1972, p.76], which requires that the computed solution be "near the exact solution of a slightly perturbed problem." Datta (1995, p.87) has called such stability *mild-stability.* In the above proof, the quantity $\bar{\gamma}e_n^t E$ is the difference between the computed solution and the exact solution for the perturbed problem, and with $k \equiv k(\beta + \delta\beta, H + \delta H)$ we have (pessimistically) $\|\bar{k} - k\|/\|k\| \leq cn^2\boldsymbol{u}$, where c is a constant of order unity.

Theorem 5.2 *The following 3-step procedure for solving the controllable single-input EAP for* (A, b, Ω), *is backward stable if step 2 is backward stable.*

Step 1 *Using the method of Householder, reduce the pair* (A, b) *to the controller-Hessenberg form* $(H, r) = (QAQ^t, Qb)$.

Step 2 *Compute the solution* k *to the EAP for* (H, r, Ω).

Step 3 *Compute* $f = Q^t k$.

Proof: Let $\bar{H}$, $\bar{r}$, and $\bar{Q}$ be the computed versions of H, r, and Q respectively. There exists an orthogonal matrix $\hat{Q}$ such that

$$\hat{Q} - \bar{Q} = E_Q, \;\; \hat{Q}A\hat{Q}^t = \bar{H} + \hat{E}_H \text{ and } \hat{Q}b = \bar{r} + \hat{\epsilon}_r \tag{5.2}$$

where $\|E_Q\| \leq \boldsymbol{u}y_0(n)$, $\|\hat{E}_H\| \leq 2cn\boldsymbol{u}\|A\|$, $\hat{\epsilon}_r \leq cn\boldsymbol{u}\|b\|$ and y_0 behaves, for all practical purposes, like $cn^{3/2}$, with c of order 10 [Wilkinson (1965), 160-161].

Let $\bar{k}$ be the computed solution to the reduced problem $(\bar{H}, \bar{r}, \Omega)$. Denote by $\bar{f}$ the computed solution to the original problem; then

$$\bar{f} \equiv \mathrm{fl}(\bar{Q}^t\bar{k}) = \bar{Q}^t\bar{k} + \bar{\epsilon}_f, \tag{5.3}$$

where $\|\bar{\epsilon}_f\| \leq n\boldsymbol{u}\|\bar{k}\|$. Substituting (5.2) into (5.3), we have

$$\bar{f} = \hat{Q}^t(\bar{k} + \hat{\epsilon}_k) \tag{5.4}$$

Here $\hat{\epsilon}_k = \bar{\epsilon}_f - E_Q^t\bar{k}$, and so $\|\hat{\epsilon}_k\| \leq \boldsymbol{u}ny_1(n)\|\bar{k}\|$, $y_1(n) = y_0(n) + n$.

Now by hypothesis, there exist $(\tilde{H}, \tilde{r})$ close to $(\bar{H}, \bar{r})$, and $\tilde{k}$ close to $\bar{k}$, such that $\tilde{k}$ is the exact solution to the EAP with input $(\tilde{H}, \tilde{r}, \Omega)$. Write $\bar{H} = \tilde{H} + \tilde{E}_H$, $\bar{r} = \tilde{r} + \tilde{\epsilon}_r$ and $\bar{k} = \tilde{k} + \tilde{\epsilon}_k$, where

$$\|\tilde{E}_H\| \leq \boldsymbol{u} z_H \|A\|, \;\; \|\tilde{\epsilon}_r\| \leq \boldsymbol{u} z_r \|b\|, \text{ and } \|\tilde{\epsilon}_k\| \leq \boldsymbol{u} z_k \|\bar{k}\|.$$

Finally, taking $\delta A \equiv \hat{Q}^t(\tilde{E}_H + \hat{E}_H)\hat{Q}$, $\delta b \equiv -\hat{Q}^t(\tilde{\epsilon}_r + \hat{\epsilon}_r)$ and and $\delta \bar{f} = -\hat{Q}^t(\tilde{\epsilon}_k + \hat{\epsilon}_k)$, we have

$$A + \delta A - (b + \delta b)(\bar{f} + \delta \bar{f})^t = \hat{Q}^t(\tilde{H} - \tilde{b}\tilde{k}^t)\hat{Q}, \tag{5.5}$$

with $\|\delta A\| \leq \boldsymbol{u}\|A\|(z_H + 2cn)$, $\|\delta b\| \leq \boldsymbol{u}\|b\|(z_r + cn)$, and $\|\delta f\| \leq \boldsymbol{u}\|\bar{k}\|(z_k + y_1(n))$.

Remark: Since our proposed Hessenberg algorithm is backward stable, the above theorem guarantees that our method is backward stable for the pair (A, b).

5.2 An Error Analysis of the Recursive Single-Input Method

Recall that this method computes a matrix L and a vector k such that $HL - L\Lambda = ke_n^t$. A careful look at the iteration reveals that the forward error has a special form. Define the polynomials $\phi_{j,k}$ for $j \leq k$ by

$$\phi_{j,k}(x) = (x - \lambda_j)(x - \lambda_{j+1}) \cdots (x - \lambda_k).$$

Theorem 5.3 *Let $\bar{\alpha}\bar{f}$ be the* computed *solution to the single-input eigenvalue assignment problem for $(H, \beta e_1, \Omega)$. If αf is the exact solution, then*

$$\bar{\alpha}\bar{f} - \alpha f = \sum_{j=1}^{n} \phi_{j,n}(H)\epsilon_j. \tag{5.6}$$

Proof: Let $\bar{l}_i$ be the computed value of the i^{th} column of L. Define ϵ_i by $\bar{l}_{i+1} = (H - \lambda_i I)\bar{l}_i + \epsilon_i$. Since $\bar{l}_1 = l_1$, we must have that $\bar{l}_2 = l_2 + \epsilon_2$; suppose $\bar{l}_i = l_i + \sum_{j=1}^{i-1} \phi_{j,i-1}(H)\epsilon_j$. Then

$$\begin{aligned}
\bar{l}_{i+1} &= (H - \lambda_i I)\bar{l}_i + \epsilon_i \\
&= (H - \lambda_i)(l_i + \textstyle\sum_{j=1}^{i-1} \epsilon_j \phi_{j,i-1}(H)) + \epsilon_i \\
&= l_{i+1} + \textstyle\sum_{j=1}^{i} \phi_{j,i}(H))\epsilon_j.
\end{aligned}$$

Now $\bar{\alpha}\bar{f} = \bar{l}_{n+1}$, and therefore

$$\bar{\alpha}\bar{f} = l_{n+1} + \sum_{j=1}^{n} \phi_{j,n}(H))\epsilon_j,$$

or

$$\bar{\alpha}\bar{f} - \alpha f = \sum_{j=1}^{n} \phi_{j,n}(H))\epsilon_j.$$

The ϵ_j can easily be bounded; for example if a machine base scaling is used to normalize $\bar{l}_j$, then it is simple to show that

$$||\epsilon_j||_F \leq \beta_m n \boldsymbol{u} ||H - \lambda_j||_F,$$

where β_m is the base. Unfortunately, not much can be said about backward stability from a result like this. It is not a necessarily bad result either, for the closed-form expression for the single-input feedback is $\alpha e_n^t \phi_{1,n}(H)$.

It is possible to shed some light on the stability of this method by looking at the ϵ_j in a different way.

Theorem 5.4 *Let $E = [\epsilon_1, \epsilon_2, \ldots, \epsilon_n]$ and let $\bar{L} = [\bar{l}_1, \bar{l}_2 \ldots, \bar{l}_n]$. Then $\bar{\alpha}\bar{f}$ solves (exactly) the single-input EAP for the perturbed system $(H - E\bar{L}^{-1}, \beta e_1, \Omega)$, where the ϵ_i are the same as in theorem 5.3.*

Proof: Notice that as defined $\bar{L}$ satisfies the Sylvester equation

$$H\bar{L} - \bar{L}\Lambda = E + \bar{\alpha}\bar{f}e_n^t,$$

where $\Lambda = diag(\lambda_i)$. Since $\bar{L}$ is nonsingular by construction, we can solve the perturbed equation

$$(H + \Delta H)\bar{L} - \bar{L}\Lambda = E + \bar{\alpha}\bar{f}e_n^t \tag{5.7}$$

for ΔH. This yields $-\Delta H = E\bar{L}^{-1}$, and by satisfying (5.7), $\bar{\alpha}\bar{f}$ solves the EAP for $(H + \Delta H, \beta e_1, \Omega)$.

5.3 Remarks on Numerical Stability and Reliability

¿From the above result we cannot say that the method is backward stable. We have simply provided an upper bound on the size of the ball around the initial data, inside which there exist $(H + \Delta H, \beta + \delta\beta)$ for which the computed solution is exact. If $||\Delta H||$ could be bounded above by a small quantity that was relatively independent of the initial data, then the method would be backward stable. But Theorem 5.4 does allow one to say precisely when the results from the method are suspect. It is clear that $||E||$ is always small if the iterates are normalized every few steps, so that all of the backward error information is contained in $\bar{L}^{-1}$. Since $\bar{L}$ is triangular, it is possible to estimate $||\bar{L}^{-1}||$ rather cheaply, even as the iteration proceeds.

The matrix L yields a bit more information about the eigenvalue assignment problem. If the closed-loop eigenvalue problem is poorly conditioned, then we cannot expect the closed-loop eigenvalues to be correct (or even well-defined), even when the feedback f is computed to very high accuracy. Now we know from construction that the method yields matrices L and Λ such that

$$H - \beta e_1 f^t = L\Lambda L^{-1},$$

where Λ is bidiagonal. It is easy to show that if the closed-loop eigenvalues are distinct, then Λ is diagonalized by the matrix $X = [x_{ij}]$, where

$$x_{ij} = \begin{cases} 1, & i = j \\ \prod_{k=1}^{j-1}(\lambda_k - \lambda_j)^{-1}, & j > i \\ 0, & j < i \end{cases}$$

Therefore, the closed-loop matrix is diagonalized by the matrix $P = L^{-1}X$ which is conveniently factorized into triangular factors, with X a unit upper triangular matrix. The inverse of P is given by $P^{-1} = X^{-1}L$, where $X^{-1} = [y_{ij}]$, and

$$y_{ij} = \begin{cases} 1, & i = j \\ \prod_{k=i+1}^{j}(\lambda_j - \lambda_k)^{-1}, & j > i \\ 0, & j < i \end{cases}$$

This leads us to an upper bound on the eigencondition of the closed-loop matrix

$$\|P\|\|P^{-1}\| = \|L^{-1}X\|\|X^{-1}L\| \leq \|X\|\|X^{-1}\|\|L\|\|L^{-1}\|.$$

The triangular factors facilitate an $O(n^2)$ LINPACK-like condition estimator of $P = L^{-1}X$. We cannot say that whenever L is illconditioned, the closed-loop eigenvalues are ill-conditioned, for L is simply a factor of P; but computational experience has shown that it is a good indicator.

Numerical Experiments

We include here several computational experiments that compare the accuracy of the proposed method (RQ) with that of Miminis & Paige (MP) and Datta. The M&P method was chosen as representative of the QR-based methods primarily because of the matlab script SEVAS, written by Miminis, and available to the public (Miminis 1991). All computations were done on a Sun Sparcstation LX. Matlab, version 4.2C, was used to compute the feedback vector using the m-files SEVAS.m for the MP method, SIPPD.m for Datta's method, and SIPPRQ.m for the proposed method (SIPPD.m and SIPPRQ.m available from Arnold). Matlab computations are double precision with a machine epsilon of $\mu = 2^{-52}$. In all tests an "exact" feedback was computed using the method of Datta, coded in D. Bailey's multi-precision fortran (1992) with a 500 decimal digit floating point representation. Datta's method was chosen for its efficiency and ease of implementation. In all of the experiments, the initial data is in controller-Hessenberg form. The computation of an exact solution allows one to avoid the eigenvalue computation (and the associated errors) necessary in the common practice of measuring error by computing the eigenvalues of H, removing its first row, and then assigning the original eigenvalues to the perturbed matrix.

For a backward stable method, one expects the size of the error in the computed solution to be roughly equal to the product of the machine epsilon and the condition number of the problem. We have included in these tests the computation of a relative condition estimator (the estimator ν_ϕ, given in (Arnold 1992), requires about $\frac{1}{15}$ the work of either of the methods being compared). We would like to emphasize two points here: first, we are measuring the error in the computed feedback, not in the closed-loop eigenvalues; and second, this condition estimator is neither a lower nor upper bound on the true condition number, which, while computable, requires at least $O(n^4)$ flops for the general case.

For all of these experiments the Matlab code that generates the test data, and the seeds for the random number generator are available from Arnold.

In the first experiment, a random matrix with elements uniformly distributed in $[-1, 1]$ was generated using Matlab's RAND function. This matrix was then reduced to Hessenberg form and its elements rounded to 15 binary digits, resulting in the system matrix H. Next, a unit random vector, r, was generated and the eigenvalues of the matrix $H - e_1 r^t$ computed. These eigenvalues, rounded to 15 binary digits, become the desired closed-loop poles. For a relatively well conditioned eigenvalue assignment problem, we expect the exact feedback to have norm near unity.

Thirty such runs were performed on matrices of size $n = 100$. The results are decribed in figure 1 and table 1. Figure 1 is a scatter plot showing $-\log_{10}(e_c)$, where $e_c = \|f - f_c\|/\|f\|$, f is the exact feedback and f_c is the feedback computed by one of the methods being compared. The x-axis serves only to label the data points; each integer k, from 1 to 30 represents a data point, and each data point consists of 4 quantities, namely the predicted error and the error for each of the 3 methods being compared. The y-axis in the figure represents the (negative of) the number of correct digits in the computation, thus a smaller (closer to $-\infty$) y-component, represents a smaller error. In order to make the plot easier to read the data is sorted by the predicted error, $\mu\nu_\phi$, and the predicted error is plotted as a continuous curve by linear interpolation. Note that even for problems of size $n = 100$ (considered large for single-input eigenvalue assignment), the feedback vector is computed to high relatively accuracy by the backward stable methods. This observation supports the argument that the generically dismal behavior of eigenvalue assignment for large n is not caused by innacurate feedback, but is primarily attributable to the conditioning of the closed-loop eigenvalues relative to the size of the feedback vector.

Table 1 provides some statistics associated with the data shown in figure 1. The quantity "digits accurate" is simply $-\log_{10}(e_c)$, which is approximately the number of correct decimal digits. The least accurate result in the sample is reported under "Minimum" accurate digits, and the average number of correct digits in the sample is reported under "Average". In an attempt to remove the "bias" of conditioning from the statistics, a backward error statistic is also computed as $e_b = e_f/\nu_{d\phi}$. The justification for this statistic is that given a backward stable method, the *true condition number* ν, and a small (relative to $1/\nu$) machine epsilon μ, the quantity e_f/ν should be approximately bounded by μ. Thus, we define the quantity "backward digits accurate" as $-\log_{10}(e_b)$. The least accurate sample with respect to this scaled error is reported as "Backward Minimum", and the average of the scaled errors is reported as "Backward Average".

Table 1 Summary Statistics of Relative Errors for 30 Randomly Generated Systems of order 100

	Accurate Digits			
Method	*Average*	*Minimum*	*Backward Average*	*Backward Minimum*
M&P	12.5	11.9	16.0	15.1
RQ	12.7	11.9	16.1	15.6
Datta	8.33	7.30	11.8	10.6

The next experiment is constructed as the first, but with $n = 20$, and with illconditioning introduced by uniformly scaling the subdiagonal entries of H so that the product of these entries is between 1×10^{-10} and unity. Again, we include a scatter plot for 100 runs, and a table summarizing the results; these are given in figure 2 and table 2, respectively.

Table 2 Summary Statistics of Relative Errors for 100 Randomly Generated Systems of order 20 of Varying Degrees of Illconditioning

	Accurate Digits			
Method	*Minimum*	*Average*	*Backward Average*	*Backward Minimum*
M&P	5.31	10.8	16.0	14.5
RQ	5.50	10.8	16.0	14.5
Datta	5.21	10.8	16.0	13.9

The last experiment is constructed as the first, but with the Hessenberg matrix H always set to (see Miminis 1981)

$$H = \begin{bmatrix} -1 & -1 & -1 & \cdots & -1 \\ 1 & -1 & -1 & \cdots & -1 \\ 0 & 1 & -1 & \cdots & -1 \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & 1 & 1 \end{bmatrix}.$$

A perturbation on the order of 2^{1-n} makes this system uncontrollable. The system size varied from $n = 3$ to $n = 32$, and one sample was taken for each n. In figure 3, we display the errors as a function of n, and as such, the data are not sorted.

Table 3 Summary Statistics of Relative Errors for Example 3 of orders 3 to 32

	Accurate Digits			
Method	*Average*	*Minimum*	*Backward Average*	*Backward Minimum*
M&P	8.52	1.22	16.1	15.5
RQ	8.48	1.67	16.1	15.4
Datta	9.16	2.18	16.7	15.5

Summary and Conclusions

In this paper, we have considered various computational aspects of the single-input eigenvalue assignment problem in control theory. We summarize the results of the paper below.

I. We have built a framework around which the QR-based methods are all special cases. We have found that these apparently different methods differ only on how the RQ decompositions are computed.

II. We have proposed a new method based on the RQ formulation of the recursive algorithm of Datta (1987). An intimate relationship of the latter with the other QR methods has been exposed via an explicit formula of the feedback vector obtained from the recursive algorithm.

III. We have proved that the proposed algorithm is backward stable by a round-off error analysis. A more general theorem obtained in this context shows that an algorithm is backward stable if the associated Hessenberg-algorithm is stable. It remains to see if the stability of the other QR algorithms can be reproved from the relationship mentioned in I.

IV. We have given a detailed round-off error analysis of the recursive algorithm. Our analysis shows precisely when the results are suspect, and this phenomenon can be determined as the algorithm proceeds, in a relatively inexpensive way.

V. We have reported the results of a numerical comparison of some of the methods.

In the multi-input case, even though the solution is not unique, it might still be possible to obtain a relationship between the solutions obtained by different algorithms. A parameterized expression for the closed-form solution obtained in the Thesis of Arnold (1992) might play an important role in this context. Also, a QR formulation of the multi-input algorithm in Arnold and Datta (1990) is in order.

References

[1] Ackermann J. *Der Entwurf Linear Regelungssysteme im Zustandsraum.* Regulungestechnik und Prozessedatenverarbeitung, vol. 7, 297-300, 1972

[2] Arnold, M. *Algorithms and Conditioning of the Eigenvalue Assignment Problem,* Ph.D. Dissertation, Northern Illinois University, DeKalb, December 1992.

[3] Arnold M. and B.N. Datta *An Algorithm for the Multiinput Eigenvalue Assignment Problem.* IEEE Trans. Automat. Contr., vol. 35, no. 10, 1149-1152, 1990

[4] Bhattacharyya S.P. and DeSouza E. *Pole assignment via Sylvester's equation*, Syst. Contr. Letter., vol. 1, pp. 261-283, 1982.

[5] Bailey, D.H. *A Portable High Performance Multiprecision Package.* RNR Technical Report RNR-90-022, 1992

[6] Bailey, D.H. *Automatic Translation of Fortran Programs to Multiprecision.* RNR Technical Report RNR-91-025, 1992

[7] Boley D.L. and G.H. Golub *The Lanczos-Arnoldi Algorithm and Controllability.* Systems and Control Letters, vol. 4, no. 6, 317-324, 1984

[8] Boley D.L. *Computing the Controllability/Observability Decomposition of a Linear Time-Invariant Dynamic System, A Numerical Approach.* Ph.D. Thesis, Stanford University, Dept. of Computer Science, Report no. STAN-CS-81-860, 1981

[9] Bru R., J. Mas and A. Urbano *An Algorithm for the Single Input Pole Assignment Problem.* SIAM J. Matrix and Appl. (1994a), 393-407.

[10] R. Bru, J. Cerdan, and A. Urbano, *An Algorithm for the Multi-input Pole Assignment Problem,* Lin. Alg. Appl. (1994b).

[11] R. Bru, J. Cerdan, P. Fernandez de Cordoba, and A. Urbano, *A Parallel Algorithm for the Partial Single-input Pole Assignment Problem*, Appl. Math. letters, Vol. 7 (1994c), 7-11.

[12] Chen, C.-T. *Linear System Theory and Design.* CBS College Publishing, New York, 1984

[13] E. Chu and B.N. Datta, *Numerically Robust Pole Assignment for Second-Order Systems*, Accepted in Int. J. Control.

[14] Cox C.L. and W.F. Moss *Backward Error Analysis for a Pole Assignment Algorithm.* SIAM J. Matrix Anal. Appl., vol. 10, no. 4, 446-456, 1989

[15] Cox C.L. and W.F. Moss *Backward Error Analysis of a Pole Assignment Algorithm II: The Complex Case.* SIAM J. Matrix Anal. Appl., vol. 13, 1159-1171, 1992

[16] M.G. Coutinho, A. Bhaya, B.N. Datta *Parallel algorithm for the eigenvalue assignment problem in linear systems*, submitted.

[17] Datta B.N. and K. Datta, *Efficient parallel algorithms for controllability and eigenvalue assignment problems,* Proceedings of the 25th IEEE Conference on Decision and Control, Athens, Greece, (1986), 1611-1616.

[18] Datta B.N. *An Algorithm to Assign Eigenvalues in a Hessenberg Matrix: Single Input Case.* IEEE Trans. Automat. Contr., vol. AC-32, no. 5, 414-417, 1987

[19] Datta B.N. and K. Datta *On Eigenvalue and Canonical Form Assignments.* Linear Algebra Appl. 131:161-182, 1990

[20] Datta B.N., *Parallel algorithms in control theory*, an invited paper in the Proceedings of the IEEE Conference on Decision and Control, (1991), 1700-1704.

[21] Datta B.N., *Large-Scale and Parallel Matrix Computations in Control: A Tutorial* (invited paper), Proceedings of the American Control Conference, June 1992.

[22] Datta B.N. *Observer Matrix Equations and the Eigenvalue Assignment Problem.* Unpublished Manuscript, 1992.

[23] Datta K. *The Matrix Equation* $XA - BX = R$ *and Its Applications.* Linear Algebra Appl. 109:91-105, 1988

[24] B.N. Datta, *Numerical Linear Algebra and Applications,* Brooks/Cole Publishing Company, Pacific Grove, California, 1995.

[25] B.N. Datta, S. Elhay, and Y. Ram, *Orthogonality and Partial Pole Assignment for the Symmetric Definite Pencil*, accepted, Lin. Alg. Appl.

[26] B.N. Datta and Y. Saad, *Arnoldi Methods for Large Sylvester-like Matrix Equations and an Associated Algorithm for Partial Spectrum Assignment*, Lin. Alg. Appl. 154-156 (1991), 225-244.

[27] Golub G.H. and C. Van Loan *Matrix Computations.* Second Edition, John Hopkins University Press, Baltimore, 1989

[28] Kailath T. *Linear Systems.* Prentice-Hall, Englewood Cliffs, N.J., 1980

[29] Kautsky J., N.K. Nichols and P. Van Dooren *Robust Pole Assignment in Linear State Feedback.* Int. J. Control, vol. 41, no. 5, 1129-1155, 1985

[30] Keel L.H., Fleming J.A., and Bhattacharyya S.P. *Pole Assignment Via Sylvester's Equation.* Contemp. Math., vol. 47, 265, 1985

[31] MathWorks, Inc., The *MATLAB User's Guide.* The MathWorks, Inc., South Natick, MA, 1992

[32] Miminis G.S. and C.C. Paige *An Algorithm for Pole Assignment of Time Invariant Linear Systems.* Int. J. Control, vol. 35, no. 2, 341-354, 1982

[33] Miminis G.S. *Numerical Algorithms for Controllability and Eigenvalue Allocation.* Ph.D. thesis, School of Computer Science, McGill University, 1981

[34] Miminis G.S. and C.C. Paige *A Direct Algorithm for Pole Assignment of Time-Invariant Multi-Input Linear Systems Using State Feedback.* Automatica, vol. 24, no. 3, 343-356, 1988

[35] Miminis, G.S. *Polepack.* 1991, (A collection of Matlab programs for Eigenvalue Assignment, available on netlib)

[36] Paige C.C. *Properties of Numerical Algorithms Related to Computing Controllability.* IEEE Trans. Automat. Contr. vol. AC-26, no. 1, 130-138, 1981

[37] Patel R.V. and P. Misra *Numerical Algorithms for Eigenvalue Assignment by State Feedback.* Proc. IEEE, vol. 72, no. 12, 1755-1764, 1984

[38] Petkov P.Hr., N.D. Christov and M.M. Konstantinov *A Computational Algorithm for Pole Assignment of Linear Multiinput Systems.* IEEE Trans. Automat. Contr., vol. AC-31, no. 11, 1044-1047, 1986

[39] Petkov P.Hr., 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, no. 11, 1045-1048, 1984

[40] Y. Saad, *Projection and Deflation Method for Partial Pole Assignment in Linear State Feedback*, IEEE Trans. Automat. Control Vol. AC-33 (1988), 290-297.

[41] Stewart, G.W. *Introduction to Matrix Computations.* Academic Press, Orlando, 1973

[42] Tsui C.-C. *An Algorithm for Computing State Feedback in Multiinput Linear Systems.* IEEE Trans. Automat. Contr., vol. AC-31, no. 3, 243-246, 1986

[43] F. Szidarovszky and A.T. Bahil *Linear Systems Theory*, CRC Press, Boca Raton, 1991.

[44] Van Dooren P.M. and M. Verhaegen *On the Use of Unitary State-Space Transformations.* Contemporary Mathematics, vol. 47, 447-463, 1985

[45] Varga A. *A Schur Method for Pole Assignment.* IEEE Trans. Automat. Contr., vol. AC-26, no. 2, 517-519, 1981

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

[47] Wonham, W.M. *Linear Multivariable Control: A Geometric Approach.* 2nd edition, Springer, New York, 1979

CAPTIONS FOR FIGURES:

Figure 1: Scatter plot for 30 problems of size 100. $log_{10}(e_c)$ is (the negative of) the number of correct decimal digits in the computed feedback. The predicted error is given by the "continuous" curve, Datta's method is represented by '*', MP by 'o', and the RQ method by 'x'.

Figure 2: Scatter plot for 100 problems of size 20. $log_{10}(e_c)$ is (the negative of) the number of correct decimal digits in the computed feedback. The predicted error is given by the "continuous" curve, Datta's method is represented by '*', MP by 'o', and the RQ method by 'x'.

Figure 3: Plot for 29 problems of size $n = 3$ to $n = 32$. Now the x-axis represents the size of the system, and the data has not been sorted. $log_{10}(e_c)$ is (the negative of) the number of correct decimal digits in the computed feedback. The predicted error is given by the "continuous" curve, Datta's method is represented by '*', MP by 'o', and the RQ method by 'x'.