TAIWANESE JOURNAL OF MATHEMATICS
Vol. 11, No. 5, pp. 1485-1502, December 2007
This paper is available online at http://www.math.nthu.edu.tw/tjm/

A SCHUR-NEWTON ALGORITHM FOR ROBUST POLE ASSIGNMENT

Tiexiang Li and Eric King-Wah Chu

Abstract. We propose an algorithm for the state feedback pole assignment problem. The algorithm is the first of its kind, making use of the Schur form and minimizing the departure from normality of the closed-loop poles by Newton's method. Eleven illustrative examples, comparing our algorithm with three other existing ones, are given.

1. Introduction

Let (A, B) denote the control system

$$\dot{x} = Ax + Bu \tag{1}$$

with the open-loop system matrix $A \in \mathbb{R}^{n\times n}$ and the input matrix $B \in \mathbb{R}^{n\times m}$. The state feedback pole assignment problem (SFPAP) seeks a control matrix $F \in \mathbb{R}^{m\times n}$ such that the closed-loop system matrix $A_c \equiv A + BF$ has prescribed eigenvalues or poles. Equivalently, we are seeking a control matrix F such that

$$(A + BF)X = X\Lambda \tag{2}$$

for some given Λ with desirable poles and nonsingular matrix X. Notice that Λ does not have to be in Jordan form, and X can be well-conditioned even with defective multiple eigenvalues in some well-chosen Λ. This is similar in spirit to the "synthesis problem" in [14]. The choice of Λ and X will be important in our discussion. In [5, 6], Λ has been chosen to be upper triangular but X has not been chosen to be unitary. In [19], poles have been assigned one (or one complex conjugate pair) at a time, after assigned poles have been shifted away from the lower-right corner of the (real) Schur form. In [8], from which our algorithm is

Received June 16, 2006, accepted July 12, 2006.
Communicated by Wen-Wei Lin.
2000 *Mathematics Subject Classification*: 15A22, 93B52, 93B55.
Key words and phrases: Robust pole assignment, Departure from normality measure, Schur form.

developed, (Λ, X) in Schur form has been chosen together with the upper triangular part of Λ (with norm equals the departure from normality $\Delta_\nu(A_c)$) minimized.

The SFPAP is solvable for *arbitrary* closed-loop spectrum when (A, B) is controllable, i.e., when $[sI - A, B]$ ($\forall s \in \mathbb{C}$) or $[B, AB, \cdots, A^{n-1}B]$ are full-ranked. The problem has been thoroughly investigated and everyone have their own favourite approach; (see, e.g., [5, 7, 13, 19-21] or any standard textbook in control theory). It is well known that the single-input case ($m = 1$) has a unique solution, while the multi-input case has some degrees of freedom left in the problem. A notable effort in utilizing these degrees of freedom sensibly was made by Kautsky et al. in [13], with the conditioning of the closed-loop spectrum (or actually $\|X^{-1}\|_F$ where X contains the normalized closed-loop eigenvectors) being optimized. When solving the pole assignment problem with a *particular* robustness measure optimized, we call it a robust pole assignment problem (RPAP). In this paper, we consider the SFPAP as well as the RPAP for (1) with state feedback. It is important to realize that there are *many* RPAPs, with different robustness measures. We are using the departure from normality $\Delta_\nu(A_c)$, the size of the upper triangular part of the Schur form, as our measure in our algorithm SCHUR. For other possibilities, see [7, 8] and Section 4.1. It will be difficult to compare one measure to another. Thus, it will be equally difficult to compare one method for a RPAP, associated with a particular robustness measure, with another. Consequently, we claim no supremacy for our method over others. We are merely trying to offer an alternative and we claim, judging from our analysis and numerical experience, that our method is feasible and has performance comparable to other methods.

When considering a particular control system, engineers will select a feedback strategy based on a lot more than just pole assignment or robustness. Our method will provide more solutions for users to choose from before applying other constraints or requirements, which are usually difficult or impossible to incorporate within the algorithms for the SFPAP or RPAP.

Various attempts have been made to minimize some measures of robustness in the RPAP. The most notable attempt was by Kautsky et al. [13], where the SFPAP in (2) was assumed to be solvable with a diagonal Λ. The problem can then be solved with the eigenvector matrix X chosen so as to minimize a condition number (e.g., $\kappa_F(X) = \|X\|_F\|X^{-1}\|_F$, or equivalently $\|X^{-1}\|_F$ with the eigenvectors x_j in X normalized). The minimization processes (for different robustness measures) are iterative, involve updating individual eigenvectors one (or two) at a time. Typically (and similar to other comparable methods), updating one eigenvector involves $O(n^3)$ flops and one sweep of all n eigenvectors requires $O(n^4)$ flops, with many sweeps required for some badly behaved examples. This can make the general approach relatively expensive. We may only want to find an acceptable suboptimal solution without going through a full iterative optimization process, as in [8]. Our algorithm

is developed from the one in [8] with full optimization (by Newton's iteration) of the departure from normality measure of the close-loop system matrix A_c.

In [19], the open-looped system is transformed into a real Schur form. The lower-right corner is then reassigned using state feedback. The assigned block is then shifted up along the diagonal and other poles are then assigned in succession. In each iterative step, the partial assignment is under-determined and the degrees of freedom can be used to minimize some robustness measure.

As mentioned earlier, there are many different possibilities in measuring robustness [7, 20]. Others prefer to minimize the size of the feedback matrix F (the feedback gain) directly. Various optimization techniques can be applied directly to these robustness or gain measures. See also the interesting approach using gradient flow [12, 15].

As far as we know, the Schur form is only utilized directly and fully in [8, 19, 20]. Our approach chooses the closed-loop Schur form directly, without using the indirect approach of partial assignment and shifting of poles in [19]. The algorithm `SCHUR` in [8] is a primitive version of our algorithm `SCHUR-NEWTON`. In [8], `SCHUR` minimized $\Delta_F(A_c)$ using generalized singular value decompositions for a given Schur vector x_1 (the first column of X in (2)), and `O-SCHUR` selected x_1 optimally using the `MATLAB` Optimization toolbox [16]. Note that the conditioning or solvability of the SFPAP is independent of the (controllable) open-loop system, as one can always apply a preliminary feedback, and we should try to avoid using the (possibly ill-conditioned) open-loop system. Another feature of the Schur form is that we do not have to restrict the structure of the eigenvalues. This allows defective eigenvalues which are ill-conditioned, usually avoided but occasionally required, as in deadbeat control.

In Section 2, we discuss briefly the perturbation results in terms of the departure from normality measure. Our Schur-Newton method is developed in Section 3. Eleven illustrative numerical examples are tested under eight robustness measures, and the numerical results are presented in Section 4. Some concluding comments are included in Section 5.

2. Departure from Normality

Consider the Schur decomposition $A_c = A + BF = X\Lambda X^\top$ with $\Lambda = D + N$ where N is the off-diagonal part of Λ. The departure of normality measure $\Delta_\nu(A) \equiv \|N\|_\nu$ was first considered by Henrici [11] and the following perturbation result was produced:

Theorem 2.1. (Henrici Theorem [11]). *Let $A,\ E \in \mathbb{C}^{n\times n}$, $E \neq 0$, $\mu \in \lambda(A+E)$ and let $\|\cdot\|_\nu$ be any norm stronger than the spectral norm (with $\|M\|_2 \leq \|M\|_\nu$ for all M). Then*

$$\min_{\lambda\in\lambda(A)} |\lambda-\mu| \le \frac{\eta}{g(\eta)}||E||_\nu \ , \quad \eta = \frac{\Delta_\nu(A)}{||E||_\nu}$$

where $g(\eta)$ *is the only positive root of* $g + g^2 + \cdots + g^n = \eta$ $(\eta \ge 0)$.

Other related perturbation results involving the departure from normality measure $\Delta_\nu(A)$ can be found in [1-4, 9, 17].

Consequently, $\Delta_\nu(A)$ can be used as a robustness measure for the close-loop spectrum, or when the conditioning of the eigenvalues of A_c has to be controlled, as in the RPAP.

3. The Schur-newton Algorithm

Consider the closed-loop eigenvalue equation (2):

$$(A+BF)X = X\Lambda$$

assuming without loss of generality that the feedback matrix B has full rank and possesses the QR decomposition

$$B = [Q_1, Q_2]\begin{bmatrix} R_B \\ 0 \end{bmatrix}. \tag{3}$$

Pre-multiplying the eigenvalue equation (2), respectively, by $B^\dagger = R_B^{-1}Q_1^\top$ and $Q_2^\top$, we obtain

$$Q_2^\top(AX - X\Lambda) = 0 \tag{4}$$

and

$$F = R_B^{-1}Q_1^\top(X\Lambda X^{-1} - A). \tag{5}$$

For a given Λ, we can select X from

$$[I_n \otimes (Q_2^\top A) - \Lambda^\top \otimes Q_2^\top]\, v(X) = 0$$

where $\otimes$ denote the Kronecker product [9] and $v(X)$ stacks the columns of X (which is slightly different from $\mathrm{Vec}(\cdot)$ defined later). For the selected X, the solution to the SFPAP can then be obtained using (5).

3.1. Real Eigenvalues

Let us first consider the case when all the closed-loop eigenvalues are real, with $A + BF = X\Lambda X^\top$ in Schur form. Here we have $\Lambda = D + N$ with $D = \text{diag}\{\lambda_1, \cdots, \lambda_n\}$ being real, and $N = [\eta_1, \eta_2, \cdots, \eta_n]$ being strictly upper triangular. Let Q denote Q_2, then we arrive at:

Optimization Problem 1. Given $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ being full rank, $Q^\top \in \mathbb{R}^{l\times n}$ $(l = n - m)$ satisfying $Q^\top B = 0$ and $D = \text{diag}\{\lambda_1, \cdots, \lambda_n\} \in \mathbb{R}^{n\times n}$, consider

$$\min_{X,N} \|N\|_F^2$$
$$\textbf{s.t.} \begin{cases} Q^\top(AX - XD) - Q^\top XN = 0 \\ X^\top X - I = 0 \end{cases}, \quad \begin{array}{l} N \text{ is } n\times n \text{ strictly upper triangular,} \\ X \text{ is } n\times n \text{ orthogonal .} \end{array}$$

In this paper, we denote $C \oplus D = \begin{bmatrix} C & 0 \\ 0 & D \end{bmatrix}$ (where C, D need not to be square), $X = [x_1, x_2, \cdots, x_n] \in \mathbb{R}^{n\times n}$, $v(X) = [x_1^\top, x_2^\top, \cdots, x_n^\top]^\top$, $v(AXB) = (B^\top \otimes A)\, v(X)$,
$\text{Vec}(I) = [1|0, 1|0, 0, 1|\cdots|0, \cdots, 0, 1]^\top \in \mathbb{R}^{n(n+1)/2 \equiv q}$. Note that both $v(\cdot)$ and $\text{Vec}(\cdot)$ stack columns of matrices but the latter discards zeroes for strictly upper triangular matrices.

Optimization Problem 1 is equivalent to:

$$\min_{X,N} \text{Vec}(N)^\top \text{Vec}(N)$$
$$\textbf{s.t.} \begin{cases} (I \otimes Q^\top A - D^\top \otimes Q^\top - N^\top \otimes Q^\top)\, v(X) = 0 \\ d_0(X)^\top v(X) - \text{Vec}(I) = 0 \end{cases}$$

where

$$N = \begin{bmatrix} 0 & \eta_{12} & \eta_{13} & \cdots & \eta_{1n} \\ 0 & 0 & \eta_{23} & \cdots & \eta_{2n} \\ \vdots & \vdots & 0 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \eta_{n-1,n} \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n\times n}, \quad \text{Vec}(N) = \begin{bmatrix} \underline{\eta_{12}} \\ \eta_{13} \\ \underline{\eta_{23}} \\ \vdots \\ \overline{\eta_{1n}} \\ \vdots \\ \eta_{n-1,n} \end{bmatrix} \in \mathbb{R}^{n(n-1)/2 \equiv p},$$

$$\begin{aligned} &d_0(C) = c_1 \oplus [c_1, c_2] \oplus [c_1, c_2, c_3] \oplus \cdots \oplus [c_1, \ldots, c_n] \in \mathbb{R}^{kn\times q}, \\ &C = [c_1, c_2, \cdots, c_n] \in \mathbb{R}^{k\times n}. \end{aligned} \tag{6}$$

We then consider the Lagrange function of the Optimization Problem 1:

$$L(\gamma, \delta, v(X), \mathrm{Vec}(N)) = \mathrm{Vec}(N)^\top \mathrm{Vec}(N)$$
$$+ \gamma^\top (I \otimes Q^\top A - D^\top \otimes Q^\top - N^\top \otimes Q^\top)\, v(X) + \delta^\top [d_0(X)^\top v(X) - \mathrm{Vec}(I)]$$

where

$$\gamma = [\underbrace{\gamma_1^\top}_{l} \big| \underbrace{\gamma_2^\top}_{l} \big| \cdots \big| \underbrace{\gamma_n^\top}_{l}]^\top \in \mathbb{R}^{ln},$$
$$R = \left[\gamma_1, \gamma_2, \cdots, \gamma_n\right],$$
$$\delta = [\underbrace{\delta_1^\top}_{1} \big| \underbrace{\delta_2^\top}_{2} \big| \cdots \big| \underbrace{\delta_n^\top}_{n}]^\top = [\delta_{11} \big| \delta_{21}, \delta_{22} \big| \cdots \big| \delta_{n1}, \delta_{n2}, \cdots, \delta_{nn}]^\top \in \mathbb{R}^q.$$

The derivatives of L satisfy

$$(7) \qquad ln \qquad \frac{\partial L}{\partial \gamma} = (I \otimes Q^\top A - D^\top \otimes Q^\top - N^\top \otimes Q^\top)\, v(X) = 0,$$

$$(8) \qquad q \qquad \frac{\partial L}{\partial \delta} = d_0(X)^\top v(X) - \mathrm{Vec}(I) = 0,$$

$$(9) \qquad n^2 \qquad \frac{\partial L}{\partial v(X)} = (I \otimes A^\top Q - D \otimes Q - N \otimes Q)\, \gamma + v(X\Delta) = 0,$$

$$(10) \qquad p \qquad \frac{\partial L}{\partial \mathrm{Vec}(N)} = 2\,\mathrm{Vec}(N) - d_1(Q^\top X)^\top \gamma = 0$$

where

$$(11) \qquad \Delta = \begin{bmatrix} 2\delta_{11} & \delta_{21} & \delta_{31} & \cdots & \delta_{n1} \\ \delta_{21} & 2\delta_{22} & \delta_{32} & \cdots & \delta_{n2} \\ \vdots & \vdots & 2\delta_{33} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & \delta_{n,n-1} \\ \delta_{n1} & \delta_{n2} & \delta_{n3} & \cdots & 2\delta_{nn} \end{bmatrix},$$

$$d_1(C) = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ c_1 \oplus [c_1, c_2] \oplus \cdots \oplus [c_1, \ldots, c_{n-1}] \end{bmatrix} \in \mathbb{R}^{kn \times p}.$$

Let

$$f_1(\gamma, \delta, v(X), \mathrm{Vec}(N)) \equiv (I \otimes Q^\top A - D^\top \otimes Q^\top - N^\top \otimes Q^\top)\, v(X) = 0,$$
$$f_2(\gamma, \delta, v(X), \mathrm{Vec}(N)) \equiv d_0(X)^\top v(X) - \mathrm{Vec}(I) = 0,$$
$$f_3(\gamma, \delta, v(X), \mathrm{Vec}(N)) \equiv (I \otimes A^\top Q - D \otimes Q - N \otimes Q)\, \gamma + v(X\Delta) = 0,$$
$$f_4(\gamma, \delta, v(X), \mathrm{Vec}(N)) \equiv 2\,\mathrm{Vec}(N) - d_1(Q^\top X)^\top \gamma = 0.$$

We can apply Newton's method to

$$f \equiv (f_1^\top, f_2^\top, f_3^\top, f_4^\top)^\top = 0$$

which can be formulated as

$$\begin{bmatrix} \gamma \\ \delta \\ v(X) \\ \mathrm{Vec}(N) \end{bmatrix}_{\mathsf{new}} = \begin{bmatrix} \gamma \\ \delta \\ v(X) \\ \mathrm{Vec}(N) \end{bmatrix} - \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial \gamma} & \frac{\partial f_1}{\partial \delta} & \frac{\partial f_1}{\partial v(X)} & \frac{\partial f_1}{\partial \mathrm{Vec}(N)} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_4}{\partial \gamma} & \frac{\partial f_4}{\partial \delta} & \frac{\partial f_4}{\partial v(X)} & \frac{\partial f_4}{\partial \mathrm{Vec}(N)} \end{bmatrix}^{-1}}_{\equiv [Jf]^{-1}} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \tag{12}$$

where the symmetric

$$Jf = \begin{bmatrix} 0 & 0 & I \otimes Q^\top A - D^\top \otimes Q^\top - N^\top \otimes Q^\top & -d_1(Q^\top X) \\ * & 0 & d_0(X)^\top + d_2(X^\top) & 0 \\ * & * & \Delta^\top \otimes I & -d_3\left(Q[\gamma_2, \cdots, \gamma_n]\right) \\ * & * & * & 2\, I_p \end{bmatrix}$$

and

$$d_2(C^\top) = \begin{bmatrix} c_1^\top \\ c_2^\top \oplus c_2^\top \\ c_3^\top \oplus c_3^\top \oplus c_3^\top \\ \vdots \\ c_n^\top \oplus \cdots \oplus c_n^\top \end{bmatrix} \in \mathbb{R}^{q \times kn}, \tag{13}$$

$$\begin{aligned} & d_3(Q[\gamma_2, \cdots, \gamma_n]) \\ & = n\{ \begin{bmatrix} Q\gamma_2 & Q\gamma_3 \oplus Q\gamma_3 & Q\gamma_4 \oplus Q\gamma_4 \oplus Q\gamma_4 & \cdots & Q\gamma_n \oplus \cdot \oplus Q\gamma_n \\ 0 & \cdots & \cdots & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{n^2 \times p}. \end{aligned} \tag{14}$$

Now, we can write down the algorithm SCHUR-NEWTON for the RPAP with real eigenvalues:

Algorithm 1.

1. Use SCHUR [8] to find the initial X_0 and N_0.
2. Substitute X_0, N_0 into (9) and (10), producing an over-determined linear system for $(\gamma_0^\top, \delta_0^\top)^\top$:

$$(15)\quad \begin{matrix} n^2 \\ p \end{matrix} \left\{ \begin{bmatrix} \overbrace{I\otimes A^\top Q - D\otimes Q - N_0\otimes Q}^{ln} & \overbrace{d_0(X)+d_2(X)^\top}^{q} \\ d_1(Q^\top X)^\top & 0 \end{bmatrix} \begin{bmatrix} \gamma_0 \\ \delta_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 2\,\mathrm{Vec}(N_0) \end{bmatrix} \right.$$

where l, p, q are defined as before and $n^2 + p \geq ln + q$.

Use the least squares method to solve the over-determined system (15) for γ_0 and δ_0.

3. With $[\gamma_0^\top, \delta_0^\top, v(X_0)^\top, \mathrm{Vec}(N_0)^\top]^\top$ being the initial starting point, run Newton's iteration (12) until convergence to X and N.
4. Substitute the X and N into (5) to obtain the feedback matrix F.

3.2. Complex Eigenvalues

When some of the closed-loop eigenvalues are complex, we assume that the given eigenvalues being $\mathcal{L} = \{\lambda_1, \cdots, \lambda_{n-2s}, \alpha_1 \pm \beta_1 i, \cdots, \alpha_s \pm \beta_s i\}$, where λ_i $(i = 1, \cdots, n-2s)$, and α_j and β_j $(j = 1, \cdots, s)$ are real. We now start from the Real Schur Decomposition $A + BF = X\Lambda X^\top$ where $\Lambda = \Omega + N$ and X is orthogonal. Here, $\Omega = D + E$ where

$$D = \mathrm{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_{n-2s}, \nu_1, \nu_2, \cdots, \nu_{2s}\} \text{ with } \begin{cases} \nu_1, \cdots, \nu_{2s} \in \mathbb{R} \\ \nu_1 + \nu_2 = 2\alpha_1 \in \mathbb{R} \\ \vdots \\ \nu_{2s-1} + \nu_{2s} = 2\alpha_s \in \mathbb{R} \end{cases},$$

$$E = 0 \oplus \cdots \oplus 0 \oplus \begin{bmatrix} 0 & \mu_2 \\ \mu_1 & 0 \end{bmatrix} \oplus \begin{bmatrix} 0 & \mu_4 \\ \mu_3 & 0 \end{bmatrix} \oplus \cdots \oplus \begin{bmatrix} 0 & \mu_{2s} \\ \mu_{2s-1} & 0 \end{bmatrix}$$

$$\text{with } \begin{cases} \nu_1\nu_2 - \mu_1\mu_2 = \alpha_1^2 + \beta_1^2 \\ \vdots \\ \nu_{2s-1}\nu_{2s} - \mu_{2s-1}\mu_{2s} = \alpha_s^2 + \beta_s^2 \end{cases}.$$

So we arrive at the optimization problem with complex eigenvalues:

Optimization Problem 2. Given $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ being full rank, $Q^\top \in \mathbb{R}^{l\times n}(l = n - m)$ satisfying $Q^\top B = 0$, $\mathcal{L} = \{\lambda_1, \cdots, \lambda_{n-2s}, \alpha_1 \pm \beta_1 i, \cdots, \alpha_s \pm \beta_s i\}$, consider

$$\min_{X,N,\Omega} \mathrm{Vec}(N)^\top \mathrm{Vec}(N)$$

$$\textbf{s.t.}\begin{cases}(I\otimes Q^\top A - D^\top\otimes Q^\top - E^\top\otimes Q^\top - N^\top\otimes Q^\top)\,v(X)=0\\ d_0(X)^\top v(X)-\mathrm{Vec}(I)=0\\ \begin{bmatrix}\nu_1+\nu_2-2\alpha_1=0\\ \vdots\\ \nu_{2s-1}+\nu_{2s}-2\alpha_s=0\end{bmatrix}\\ \begin{bmatrix}\nu_1\nu_2-\mu_1\mu_2-(\alpha_1^2+\beta_1^2)=0\\ \vdots\\ \nu_{2s-1}\nu_{2s}-\mu_{2s-1}\mu_{2s}-(\alpha_s^2+\beta_s^2)=0\end{bmatrix}\end{cases}$$

where D and E are defined as before, d_0 as defined in (6) and

$$N=\begin{bmatrix}
0 & \eta_{1,2} & \eta_{1,3} & \cdots & \eta_{1,n-2s} & \eta_{1,n-2s+1} & \eta_{1,n-2s+2} & \cdots & \eta_{1,n-1} & \eta_{1,n}\\
 & 0 & \eta_{2,3} & \cdots & \eta_{2,n-2s} & \eta_{2,n-2s+1} & \eta_{2,n-2s+2} & \cdots & \eta_{2,n-1} & \eta_{2,n}\\
 & & 0 & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
 & & & \ddots & \eta_{n-2s-1,n-2s} & \vdots & \vdots & \vdots & \vdots & \vdots\\
 & & & & 0 & \eta_{n-2s,n-2s+1} & \eta_{n-2s,n-2s+2} & \vdots & \vdots & \vdots\\
 & & & & & 0 & 0 & \vdots & \vdots & \vdots\\
 & & & & & 0 & 0 & \ddots & \vdots & \vdots\\
 & & & & & & & \ddots & \eta_{n-2,n-1} & \eta_{n-2,n}\\
 & & & & & & & & 0 & 0\\
 & & & & & & & & 0 & 0
\end{bmatrix}.$$

We then consider the Lagrange function for Optimization Problem 2:

$$\begin{aligned}
&L(\gamma,\delta,\varepsilon,\omega,v(X),\nu,\mu,\mathrm{Vec}(N))\\
&=\mathrm{Vec}(N)^\top\mathrm{Vec}(N)+\gamma^\top(I\otimes Q^\top A-D^\top\otimes Q^\top-E^\top\otimes Q^\top\\
&\quad -N^\top\otimes Q^\top)\,v(X)+\delta^\top(d_0(X)^\top v(X)-\mathrm{Vec}(I))\\
&\quad +\sum_{j=1}^{s}\varepsilon_j(\nu_{2j-1}+\nu_{2j}-2\alpha_j)+\sum_{j=1}^{s}\omega_j[\nu_{2j-1}\nu_{2j}-\mu_{2j-1}\mu_{2j}-(\alpha_j^2+\beta_j^2)]
\end{aligned}\tag{17}$$

where

$$\begin{aligned}\varepsilon&=(\varepsilon_1,\varepsilon_2\cdots,\varepsilon_s)^\top,\\ \omega&=(\omega_1,\omega_2\cdots,\omega_s)^\top,\\ \mu&=(\mu_1,\mu_2,\cdots,\mu_{2s})^\top,\end{aligned}$$

$$\nu = (\nu_1, \nu_2, \cdots, \nu_{2s})^\top,$$
$$\mathrm{Vec}(N) = (\eta_2^\top, \eta_3^\top, \cdots, \eta_{n-2s}^\top, \cdots, \eta_n^\top)^\top.$$

The derivatives of L satisfy

$$f_1 \equiv \frac{\partial L}{\partial \gamma} = (I \otimes Q^\top A - D^\top \otimes Q^\top - E^\top \otimes Q^\top - N^\top \otimes Q^\top)\, v(X) = 0, \tag{18}$$

$$f_2 \equiv \frac{\partial L}{\partial \delta} = d_0(X)^\top v(X) - \mathrm{Vec}(I) = 0, \tag{19}$$

$$f_3 \equiv \frac{\partial L}{\partial \varepsilon} = [\nu_1 + \nu_2 - 2\alpha_1, \cdots, \nu_{2s-1} + \nu_{2s} - 2\alpha_s]^\top = 0, \tag{20}$$

$$f_4 \equiv \frac{\partial L}{\partial \omega} = \begin{bmatrix} \nu_1\nu_2 - \mu_1\mu_2 - (\alpha_1^2 + \beta_1^2) = 0 \\ \vdots \\ \nu_{2s-1}\nu_{2s} - \mu_{2s-1}\mu_{2s} - (\alpha_s^2 + \beta_s^2) = 0 \end{bmatrix}, \tag{21}$$

$$f_5 \equiv \frac{\partial L}{\partial v(X)} = (I \otimes A^\top Q - D \otimes Q - E \otimes Q - N \otimes Q)\, \gamma + v(X\Delta) = 0, \tag{22}$$

$$\begin{aligned} f_6 \equiv \tfrac{\partial L}{\partial \nu} = &\begin{bmatrix} \varepsilon_1 \\ \varepsilon_1 \\ \vdots \\ \varepsilon_s \\ \varepsilon_s \end{bmatrix} + \begin{bmatrix} \omega_1\nu_2 \\ \omega_1\nu_1 \\ \vdots \\ \omega_s\nu_{2s} \\ \omega_s\nu_{2s-1} \end{bmatrix} \\ &- \begin{bmatrix} \gamma_{n-2s+1}^\top & & \\ & \ddots & \\ & & \gamma_n^\top \end{bmatrix} v(Q^\top [x_{n-2s+1}, \cdots, x_n]) = 0, \end{aligned} \tag{23}$$

$$\begin{aligned} f_7 \equiv \tfrac{\partial L}{\partial \mu} = &- \begin{bmatrix} \omega_1\mu_2 \\ \omega_1\mu_1 \\ \vdots \\ \omega_s\mu_{2s} \\ \omega_s\mu_{2s-1} \end{bmatrix} \\ &- \begin{bmatrix} \gamma_{n-2s+1}^\top & & \\ & \ddots & \\ & & \gamma_n^\top \end{bmatrix} \Pi_s\, v(Q^\top [x_{n-2s+1}, \cdots, x_n]) = 0, \end{aligned} \tag{24}$$

$$f_8 \equiv \frac{\partial L}{\partial \mathrm{Vec}(N)} = 2\,\mathrm{Vec}(N) - \widehat{d}_1(Q^\top X)^\top \gamma = 0 \tag{25}$$

where $\widehat{d}_1(C) =$

$$\begin{bmatrix} 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ c_1 \oplus \cdots \oplus & [c_1, \cdots, c_{n-2s-1}] \oplus \begin{bmatrix} c_1, \cdots, c_{n-2s} & 0 \\ 0 & c_1, \cdots, c_{n-2s} \end{bmatrix} \oplus \cdots \oplus & \begin{bmatrix} c_1, \cdots, c_{n-2} & 0 \\ 0 & c_1, \cdots, c_{n-2} \end{bmatrix} \end{bmatrix},$$

$$\Pi_s = \underbrace{\begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \oplus \cdots \oplus \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}}_{s}.$$

We can then obtain the symmetric gradient matrix of $f \equiv [f_1^\top, f_2^\top, \cdots, f_8^\top]^\top$:

$$Jf = \begin{bmatrix} 0 & 0 & 0 & 0 & \sum & -\widehat{d}_4(Q^\top X) & -d_4(Q^\top X) & -\widehat{d}_1(Q^\top X) \\ & 0 & 0 & 0 & \Xi^\top & 0 & 0 & 0 \\ & & 0 & 0 & 0 & d_5(e) & 0 & 0 \\ & & & 0 & 0 & d_5(\nu) & -d_5(\mu) & 0 \\ & & & & \Delta^\top \otimes I & -\widehat{d}_4(QR) & -d_6(QR,s) & -d_8(QR,s)^\top \\ & & * & & & d_7(\omega) & 0 & 0 \\ & & & & & & -d_7(\omega) & 0 \\ & & & & & & & 2\,I_{\widehat{p}} \end{bmatrix} \begin{matrix} \} & ln \\ \} & q \\ \} & s \\ \} & s \\ \} & n^2 \\ \} & 2s \\ \} & 2s \\ \} & \widehat{p} \end{matrix}$$

where

$$\sum = I \otimes Q^\top A - D^\top \otimes Q^\top - E^\top \otimes Q^\top - N^\top \otimes Q^\top,$$

$$\widehat{p} = (n-2s)(n-2s-1)/2 + 2s(n-s-1),$$

$$e = [\underbrace{1, 1, \cdots, 1}_{2s}]^\top \quad , \quad \Xi = d_0(X) + d_2(X^\top)^\top,$$

$$d_4(Q^\top X) = \begin{bmatrix} & 0 & & \cdots & \cdots & & 0 & \\ & \vdots & & & & & \vdots & \\ & 0 & & \cdots & \cdots & & 0 & \\ \begin{bmatrix} Q^\top x_{n-2s+2} & & Q^\top x_{n-2s+1} \end{bmatrix} & & & \oplus \cdots \oplus & & \begin{bmatrix} Q^\top x_n & & Q^\top x_{n-1} \end{bmatrix} & & \end{bmatrix} \begin{matrix} \} & l(n-2s) \\ & \\ \} & l(2s) \end{matrix},$$

$$\widehat{d}_4(Q^\top X) = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ Q^\top x_{n-2s+1} & \oplus & Q^\top x_{n-2s+2} & \oplus \cdots \oplus & Q^\top x_{n-1} & \oplus & Q^\top x_n \end{bmatrix} \begin{matrix} \} & l(n-2s) \\ & \\ \} & l(2s) \end{matrix},$$

$$d_5(\mu_1, \cdots, \mu_{2s}) \equiv d_5(\mu^\top) = [\mu_2, \mu_1] \oplus [\mu_4, \mu_3] \oplus \cdots \oplus [\mu_{2s}, u_{2s-1}],$$

$$d_6(QR,s)=\begin{bmatrix} & 0 & \cdots & \cdots & 0 & \\ & \vdots & & & \vdots & \\ & 0 & \cdots & \cdots & 0 & \\ \begin{bmatrix} 0 & Q\gamma_{n-2s+2} \\ Q\gamma_{n-2s+1} & 0 \end{bmatrix} & & \oplus\cdots\oplus & & \begin{bmatrix} 0 & Q\gamma_n \\ Q\gamma_{n-1} & 0 \end{bmatrix} \end{bmatrix} \begin{matrix} \} & l(n-2s) \\ & \\ & \\ \} & l(2s) \end{matrix},$$

$$d_7(\varepsilon)=\begin{bmatrix} 0 & \varepsilon_1 \\ \varepsilon_1 & 0 \end{bmatrix}\oplus\begin{bmatrix} 0 & \varepsilon_2 \\ \varepsilon_2 & 0 \end{bmatrix}\oplus\cdots\oplus\begin{bmatrix} 0 & \varepsilon_s \\ \varepsilon_s & 0 \end{bmatrix},$$

$$d_8(QR,s)=\begin{bmatrix} (Q\gamma_2)^\top & & & 0 \\ (Q\gamma_3)^\top\oplus(Q\gamma_3)^\top & & & 0 \\ (Q\gamma_4)^\top\oplus(Q\gamma_4)^\top\oplus(Q\gamma_4)^\top & & & 0 \\ & & \vdots & \\ (Q\gamma_{n-2s})^\top\oplus\cdots\oplus & & Q\gamma_{n-2s})^\top & 0 \\ (Q\gamma_{n-2s+1})^\top\oplus\cdots\oplus(Q\gamma_{n-2s+1})^\top & & & 0 \\ (Q\gamma_{n-2s+2})^\top\oplus\cdots\oplus(Q\gamma_{n-2s+2})^\top & & & 0 \\ & & \vdots & \\ (Q\gamma_{n-1})^\top\oplus\cdots\oplus(Q\gamma_{n-1})^\top & & & 0 \\ (Q\gamma_n)^\top\oplus\cdots\oplus & & (Q\gamma_n)^\top & 0 \end{bmatrix} \begin{matrix} \} & 1 \\ \} & 2 \\ \} & 3 \\ & \\ & \\ \} & n-2s-1 \\ \} & n-2s \\ \} & n-2s \\ & \\ & \\ \} & n-2 \\ \} & n-2 \end{matrix}.$$

Modifying Algorithm 1, we solve Optimization Problem 2 by Newton's iteration to obtain a real feedback matrix F:

Algorithm 2.

1. Use `Schur` [8] to find the initial X_0 and N_0.
2. Substitute X_0, N_0 into (20-24) and (25), we obtain $[\gamma_0^\top, \delta_0^\top, \varepsilon_0^\top, \omega_0^\top, \nu_0^\top, \mu_0^\top]^\top$ for the Newton iterations.
3. Let $[\gamma_0^\top, \delta_0^\top, \varepsilon_0^\top, \omega_0^\top, v(X_0)^\top, \nu_0^\top, \mu_0^\top, \mathrm{Vec}(N_0)^\top]^\top$ be the initial guess, apply Newton's iteration (12) until convergence to X, N.
4. Substitute the X, N and Λ into (5) to obtain the feedback matrix F.

A few remarks are in order:

- At step 2, we set $\nu_0=[\alpha_1,\alpha_1,\cdots,\alpha_s,\alpha_s]^\top$ and $\mu_0=[\beta_1,-\beta_1,\cdots,\beta_s,-\beta_s]^\top$, and obtain γ_0 by substituting X_0, N_0 into (25). Then from (22-24), we obtain δ_0, ε_0 and ω_0.
- At step 3, the initial point $[\gamma_0^\top, \delta_0^\top, \varepsilon_0^\top, \omega_0^\top, v(X_0)^\top, \nu_0^\top, \mu_0^\top, \mathrm{Vec}(N_0)^\top]^\top$ is often far away from the optimal point. In such an event, we apply the GBB Gradient method [10] to decrease the objective function sufficiently, before Newton's iteration is applied.

- At step 4, since the matrix X is orthogonal, we can use $X^\top$ in place of X^{-1}.

These remarks also hold for Algorithm 1.

4. Numerical Examples

We shall compare our method with `O-SCHUR`, `PLACE` (the MATLAB command implementing method 1 in [13]) and `RONPOLE` [18], in terms of eight different robustness measures for eleven examples (EX1-11, quoted from [8] but appeared elsewhere earlier). It is important to keep in mind that comparing different methods, each designed for a particular robustness measure, is difficult.

4.1. Robustness Measures

For each test examples, the following condition numbers were used to investigate the finial closed-loop system, with the real feedback matrix F:

1. (Condition number) $\kappa(X) = \|X\|_F\|X^{-1}\|_F$, where the columns of X contains the normalized closed-loop eigenvectors of the closed-loop system matrix A_c. The condition number for closed-loop eigenvalues has been commonly used as a robustness measure for feedback pole assignment problems.
2. (Feedback gain) $\|F\|_F$, favoured by some engineers, representing the amount of energy required for the corresponding control action.
3. (Entropy) $E \equiv \frac{1}{2}\|X\|_F^2 - \ln|\det X|$, closed related to ω below and is convenient to work with as X^{-1} is not present explicitly.
4. (Byers-Nash measure) $BN \equiv \sqrt{\|X\|_F^2 + \|X^{-1}\|_F^2}$.
5. $\omega \equiv \|X\|_F/(\sqrt{n}|\det \mathrm{X}|^{1/n})$, the square-root of the ratio between the arithmetic and geometric means of the singular values of X, with its logarithm obviously related to E.
6. $\sigma \equiv |\lambda(X)|/(|\det \mathrm{X}|^{1/n})$, the ratio between the largest singular value and the geometric mean of the singular values of X.
7. $\tau \equiv \|X\|_F/(\sqrt{n}|\lambda_n(X)|)$, the ratio between the arithmetic mean of the singular values and the smallest singular value of X.
8. (Departure from normality) $\Delta_F(A_c) = \|N\|_F$, the robustness measure for `SCHUR` and our `SCHUR-NEWTON` method.

Note that the robustness measure for `PLACE` [13] is $\|X^{-1}\|_F$ and for `ROBPOLE` [18] is $-|\det(X)|$, which is closely related to the measures involving determinants (in (3), (5) and (6) in the above list).

4.2. Numerical Results

We have applied Algorithm SCHUR [8] with u_1 chosen to be ONES(m,1) (the m-vector filled with ones), and applied PLACE and ROBPOLE with I_n as the initial matrix. Then we use the result of Algorithm SCHUR as the initial guess in SCHUR-NEWTON. The convergence tolerance is 10^{-5}. The numerical computations were carried out on a MATLAB 7.01 [16] with machine accuracy equals 2.22×10^{-16}.

Tables 1-4 have eleven rows (for the eleven examples) and nine columns (for the eight robustness measures and the cputime).

Table 1 contains the robustness measures of the closed-loop systems for the eleven examples and the CPU-times after applying SCHUR-NEWTON.

Table 1. m(SCHUR-NEWTON)

Ex	κ_F	$\|F\|_F$	E	BN	ω	σ	τ	$\|N\|_F$	cputime
1	7.2	1.4	3.3	4.2	1.3	1.2	1.9	1.2(1)	1.5(-2)
2	5.5(1)	2.8(2)	7.0	2.4(1)	2.4	2.1	6.9	2.6(1)	5.0(-1)
3	1.1(2)	3.5(1)	8.7	5.5(1)	5.4	5.6	3.1(1)	1.8(1)	1.5(-2)
4	1.3(1)	9.4	3.1	7.9	1.7	2.2	9.0(-1)	1.0(1)	1.5(-2)
5	2.3(2)	3.0	1.0(1)	1.0(2)	5.0	5.2	2.5(1)	6.8(-1)	1.5(-2)
6	1.6(1)	2.6(1)	4.6	8.6	1.9	1.4	2.5	4.9(1)	4.7(-2)
7	9.7(1)	5.4(2)	3.0	3.4(1)	1.9	1.8	1.3(1)	1.3	2.0(-1)
8	3.4(1)	4.7(1)	5.3	1.7(1)	2.3	9.0(-1)	1.2	5.5	3.1(-2)
9	7.3(2)	1.7(3)	6.7	3.6(1)	3.3	7.0(-1)	1.1	7.3	3.1(-2)
10	4.0	1.5	2.0	2.8	1.0	1.0	1.0	7.5(-5)	1.5(-2)
11	5.0(5)	1.1(3)	2.0	2.5(5)	1.2(2)	8.4(1)	3.8(4)	5.0(2)	1.8(-1)

Table 2, as suggested by label m(SCHUR-NEWTON)/m(ROBPOLE), contains the ratios of robustness measures from the feedback matrices and the CPU-times obtained respectively by SCHUR-NEWTON and ROBPOLE. Similar ratios m(SCHUR -NEWTON)/m(PLACE) and m(SCHUR-NEWTON PA)/ m(O-SCHUR) are presented, respectively, in Tables 3-4. Noting that comparing CPU-times in MATLAB is risky and Tables 1-4 only provide a rough indication of the relative speed of the algorithms, with their different implementations and varying degree of code optimization.

Table 2. Ratios m(SCHUR-NEWTON)/m(ROBPOLE)

Ex	κ_F	$\|F\|_F$	E	BN	ω	σ	τ	$\|N\|_F$	cputime
1	1.0	1.0	1.9	1.0	1.0	9.2(-1)	1.0	9.2(-1)	4.4(-2)
2	1.1	1.7	2.8	1.1	1.1	1.1	3.4(-1)	8.4(-1)	1.8
3	1.9(-1)	7.1(-1)	3.6	1.9	2.0	2.0	1.7	4.6(-1)	1.9(-1)
4	1.0	1.0	1.6	9.8(-1)	9.4(-1)	1.0	1.2(-1)	9.1(-1)	3.8(-1)
5	1.6	8.7(-1)	3.2	1.5	1.1	7.6(-1)	7.3(-1)	9.2(-1)	6.5(-2)
6	2.7	1.3	2.8	2.4	1.6	1.0	1.6	1.0	3.4(-1)
7	8.1	2.3	1.3	6.7	1.6	1.3	7.4	3.8(-1)	8.0(-2)
8	2.6	1.7	2.7	2.4	1.5	6.0(-1)	2.6(-1)	7.1(-1)	2.2(-1)
9	3.0(1)	2.0	3.1	3.0	1.8	2.6(-1)	2.2(-1)	6.1(-1)	3.9(-1)
10	1.0	1.0	1.4	1.0	1.0	1.0	1.0	9.5(-1)	1.5(-1)
11	3.3(1)	1.7(-1)	5.9(-1)	3.4(1)	1.0(1)	7.6	1.4(1)	8.9(-2)	1.0

Table 3. Ratios m(SCHUR-NEWTON)/m(PLACE)

Ex	κ_F	$\|F\|_F$	E	BN	ω	σ	τ	$\|N\|_F$	cputime
1	1.0	9.3(-1)	1.9	1.1	1.0	8.0(-1)	1.2	9.2(-1)	3.0(-1)
2	1.0	7.4(-1)	2.6	1.0	1.0	1.0	1.0	8.7(-1)	2.6
3	2.0	5.6(-1)	3.6	2.0	2.0	3.7	7.5	4.9(-1)	7.5(-1)
4	1.0	9.2(-1)	1.7	1.0	9.4(-1)	1.0	1.2(-1)	9.0(-1)	1.5
5	1.5	6.2(-1)	3.2	1.5	1.1	7.8(-1)	9.2(-1)	9.2(-1)	3.8(-1)
6	2.7	1.2	2.8	2.4	1.6	1.1	1.7	9.8(-1)	1.2
7	8.0	1.6	1.3	6.7	1.6	1.4	8.5	1.5	2.2(-1)
8	2.6	1.5	2.8	2.4	1.5	6.0(-1)	2.6(-1)	7.1(-1)	7.5(-1)
9	3.0(1)	2.0	3.2	3.0	1.8	2.5(-1)	2.2(-1)	6.1(-1)	3.1
10	1.0	1.0	1.4	1.0	1.0	1.0	1.0	8.9(-4)	5.0(-1)
11	3.3(1)	1.6(-1)	5.9(-1)	3.4(1)	1.0(1)	7.6	2.3(1)	8.8(-2)	18

Table 4. Ratios m(SCHUR-NEWTON)/m(O-SCHUR)

Ex	κ_F	$\|F\|_F$	E	BN	ω	σ	τ	$\|N\|_F$	cputime
1	1.0	1.0	1.9	1.0	1.0	7.5(-1)	7.9(-1)	9.2(-1)	7.5(-1)
2	8.5(-1)	1.1	2.6	8.2(-1)	9.6(-1)	1.1	5.7(-1)	6.7	1.6
3	2.0	4.3(-1)	3.5	1.9	1.8	3.1	5.7	3.7(-1)	1.5
4	1.0	9.6(-1)	1.7(1)	1.0	9.4(-1)	1.1	1.2(-1)	9.0(-1)	1.5
5	1.6(-1)	7.1(-1)	2.8	1.6(-1)	6.2(-1)	6.7(-1)	9.6(-2)	9.6(-1)	1.5
6	4.0(-1)	1.7	2.7	2.2	1.5	1.2	1.8	4.5	1.9(-1)
7	6.4(-1)	7.3(-1)	1.0	6.3(-1)	1.1	1.0	9.3(-1)	1.6	2.3
8	1.4	1.3	2.5	1.4	1.2	4.7(-1)	1.4(-1)	5.5(-1)	2.3
9	1.3(1)	2.0	2.9	1.3	1.5	3.7(-1)	8.4(-2)	4.3(-1)	3.1
10	1.0	1.0	1.4	1.0	1.0	1.0	1.0	8.7(-1)	1.5
11	2.0(1)	1.6(-1)	5.7(-1)	2.0(1)	9.2	6.5	1.9(1)	8.6(-2)	1.8

5. Comments on Numerical Experiments

From Tables 1-4 (and Tables 1-10 in [8]), we have the following comments:

(1) From Table 1, our SCHUR-NEWTON algorithm produced the Schur decomposition (Λ, X) efficiently. Unlike SCHUR, PLACE and ROBPOLE, which update one or two eigenvalues/eigenvectors at a time, our algorithm update all the eigenvalues and Schur vectors simultaneously.

(2) From Table 2-3, SCHUR-NEWTON performed well, comparing with ROBPOLE and PLACE. It performed generally better in terms of the departure from normality measure $\Delta_F(A_c)$, but slightly worse in the measure κ_F for most examples. Robustness measures of similar magnitudes were produced, but our algorithm required less CPU-time.

(3) From Table 4, SCHUR-NEWTON performed well comparing with O-SCHUR, especially in term of the departure from normality measure in the real-eigenvalues case. Robustness measures of similar magnitudes were produced. However, SCHUR -NEWTON required more CPU-time on most examples.

Acknowledgment

We would like to thank Professors Wen-Wei Lin and Shu-Fang Xu for various interesting discussions and much encouragement.

REFERENCES

1. C. Beattie and I. C. F. Ipsen, Inclusion regions for matrix eigenvalues, *Lin. Alg. Appl.*, **358** (2003), 281-291.
2. T. Braconnier and Y. Saad, Eigenvalue bounds from the Schur form, *Research Report*, University of Minnesota Supercomputing Institute, **UMSI 98/21** (1998).
3. G. E. Cho and I. C. F. Ipsen, If a matrix has only a single eigenvalue, how sensitive is this eigenvalue? *CRSC Technical Report*, North Carolina State University, Raleigh NC, **TR97-20** (1997).
4. G. E. Cho and I. C. F. Ipsen, If a matrix has only a single eigenvalue, how sensitive is this eigenvalue? II, *CRSC Technical Report*, North Carolina State University, Raleigh NC, **TR98-8** (1998).
5. E. K.-W. Chu, A pole assignment algorithm for linear state feedback, *Syst. Control Letts.*, **49** (1986), 289-299.
6. E. K.-W. Chu, A canonical form and a state feedback pole assignment algorithm for descriptor systems, *IEEE Trans. Automat. Control*, **AC-33** (1988), 1114-1125.
7. E. K.-W. Chu, Optimization and pole assignment in control system, *Int. J. Applied Maths. Comp. Sci.*, **11** (2001), 1035-1053.
8. E. K.-W. Chu, Pole assignment via the Schur form, *Syst. Control Letts.*, **56** (2007), 303-314.
9. G. H. Golub and C. F. van Loan, *Matrix Computations*, 2nd Ed., Johns Hopkins University Press, Baltimore, MD, 1989.
10. L. Grippo and M. Sciandrone, Nonmonotone globalization techniques for the Barzilai-Borwein gradient method, *Comput. Optim. Applics.*, **23** (2002), 143-169.
11. P. Henrici, Bounds for iterates, inverses, spectral variation and the field of values of nonnormal matrcies, *Numer. Math.*, **4** (1962), 24-40.
12. S. Hu and J. Wang, A gradient flow approach to on-line robust pole assignment for synthesizing output feedback control systems, *Automatica*, **38** (2002), 1959-1968.
13. J. Kautsky, N. K. Nichols and P. van Dooren, Robust pole assignment via in linear state feedback, *Int. J. Control*, **41** (1985), 1129-1155.
14. m. m. konstantinov, p. hr. petkov and n.d. christov, Sensitivity analysis of the feedback synthesis problem, *IEEE Trans. Autom. Control*, **AC-42** (1997), 568-573.
15. J. Lam and H. K. Tam, Pole assignment with minimum eigenvalue differential sensitivity, *Proc. Instn. Mech. Engrs.*, **211** (1997), 63-74.
16. mathworks, *MATLAB User's Guide*, 2002.
17. G. W. Stewart and J.-G. Sun, *Matrix Perturbation Theory*, Academic Press, New York, 1990.

18. A. Tits and Y. Yang, Globally convergent algorithms for robust pole assignment by state feedback, *IEEE Trans. Autom. Control*, **TAC-41** (1996), 1432-1552.

19. A. Varga, A Schur method for pole assignment, *IEEE Trans. Automat. Control*, **AC-27** (1981), 517-519.

20. A. Varga, Robust pole assignment techniques via state feedback, *Proc. CDC'2000*, Sydney, Australia, 4655-4660.

21. B. A. White, Eigenstructure assignment: a survey, *Proc. Instn. Mech. Engrs.*, **209** (1995), 1-11.

Tiexiang Li
LMAM,
School of Mathematical Sciences,
Peking University,
Beijing 100871,
P. R. China
E-mail: feco@math.pku.edu.cn

Eric King-wah Chu
School of Mathematical Sciences,
Building 28,
Monash University,
VIC 3800,
Australia
E-mail: eric.chu@sci.monash.edu.au