Using Deflation in the Pole Assignment Problem with Output Feedback

George Miminis
Department of Computer Science, Memorial University of Newfoundland
St. John's, Newfoundland, Canada A1C 5S7

Abstract

A direct algorithm is suggested for the computation of a linear output feedback for a multi input, multi output system such that the resultant closed-loop matrix has eigenvalues that include a specified set of eigenvalues. The algorithm uses deflation based on unitary similarity transformations. Thus we hope the algorithm is numerically stable, however, this has not been proven as yet.

1 Introduction

Deflation is a technique that has been efficiently used in the solution of the standard eigenvalue problem of a matrix A as well as other eigenvalue related problems. According to this technique once an eigenpair (λ_1, x_1) of A is computed, we continue the process with a matrix that possesses only the remaining eigenvalues of A, and possibly a zero eigenvalue in the place of λ_1. In this way we are left to solve a smaller problem. Deflation can be accomplished by a variety of algorithms. Some of them are, Hoteling's deflation, Wielandt's deflation, deflation based on similarity transformations, deflation by restriction, etc. An excellent review of the first three methods can be found in [7, pp. 584-600], whereas deflation by restriction can be found in [5, p. 84]. Lately, deflation has been used in the solution of eigenassignment problems. For example, Wielandt's deflation is used in [6] to solve the partial eigenvalue allocation problem with state feedback for continuous time systems. In this paper we will be concerned with deflation based on unitary similarity transformations, and how it can be used in the pole assignment problem with output feedback, or as it is often known, the eigenvalue allocation problem with output feedback (MEVAO).

In section 2 we define the MEVAO. In section 3 we define transmission zeros and discuss how they affect the MEVAO. In section 4 we give an algorithm for the solution of the MEVAO. Finally in section 5 give some numerical examples to demonstrate the performance of our algorithm.

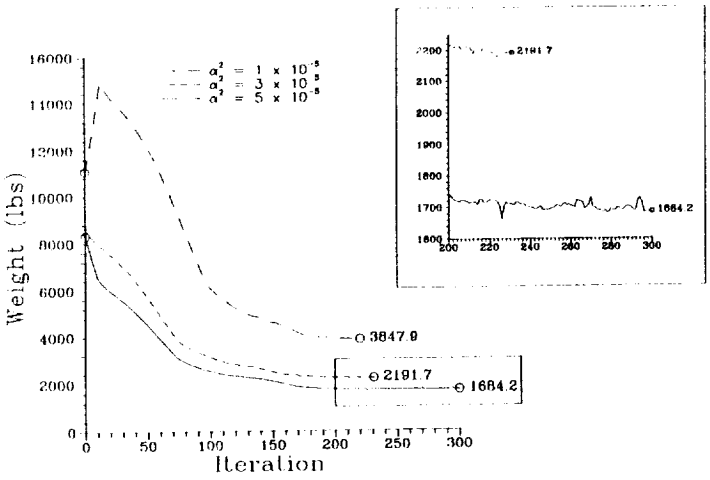

Figure 4: Convergence histories for particular cases from Table 6

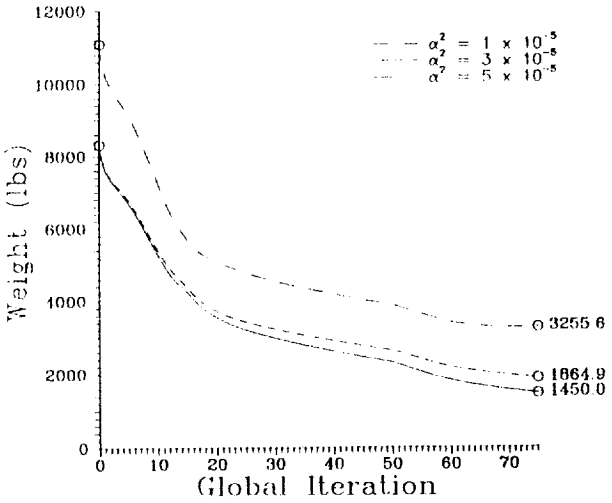

Figure 5: Convergence histories for particular cases from Table 7

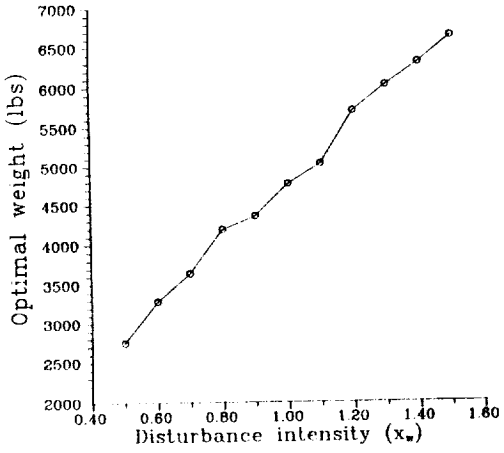

Figure 6: Optimal weight for varying external disturbance intensity $\lambda_w = x_w I$ with $\alpha^2 = 1 \times 10^{-5}$ and $\beta^2 = 50$

	α^2 ($\times 10^{-5}$)				
	1	2	3	4	5
Initial DV	120.000	90.000	90.000	90.000	90.000
Final DV					
1	92.172	55.944	41.010	44.107	36.632
2	72.274	55.967	52.340	40.128	32.776
3	71.334	62.487	41.938	38.878	31.848
4	41.867	27.830	24.475	24.192	17.445
5	47.282	33.755	24.327	19.986	18.412
6	41.794	33.553	26.973	23.608	20.363
7	41.341	2.365	5.462	17.750	8.363
8	41.314	20.562	1.604	17.277	1.317
9	2.795	20.742	23.982	1.761	10.880
10	9.518	2.424	24.159	12.659	17.091
11	9.541	24.020	1.570	14.006	20.089
12	2.795	23.903	5.612	1.667	1.458

	α^2 ($\times 10^{-5}$)				
	6	7	8	9	10
Initial DV	90.000	60.000	60.000	60.000	60.000
Final DV					
1	28.739	34.922	25.632	30.032	23.272
2	39.315	26.729	32.700	23.857	28.426
3	28.871	26.916	24.976	24.192	22.333
4	17.154	16.238	14.880	14.139	13.401
5	17.249	17.279	15.037	15.367	13.472
6	19.143	16.109	16.386	14.174	14.662
7	4.076	15.734	3.447	14.073	3.130
8	1.143	15.739	0.996	13.904	0.850
9	16.803	1.039	14.683	0.896	13.294
10	16.869	3.667	14.621	3.215	13.149
11	1.200	3.627	1.013	3.291	0.849
12	3.803	1.038	3.387	0.899	3.069

Table 10: Optimal design variables for $\beta^2 = 50$ cases from Table 9

α^2 ($\times 10^{-5}$)	initial dv's (structural)	optimal weight		iterations	
		$\beta^2 = 50$	$\beta^2 = 60$	$\beta^2 = 50$	$\beta^2 = 60$
1	120.0	3889.6	3630.7	221	247
2	90.0	2620.3	2434.5	400*	400*
3	90.0	2161.2	1975.4	400*	400*
4	90.0	1917.1	1732.4	277	400*
5	90.0	1747.7	1544.5	208	400*
6	90.0	1603.4	1541.1	199	189
7	60.0	1680.4	1365.0	160	281
8	60.0	1383.7	1234.0	187	400*
9	60.0	1252.3	1156.9	400*	400*
10	60.0	1206.3	1112.2	311	400*

* indicates no convergence in specified number of global iterations

Table 11: Optimal weight using sequential appproximations solution algorithm with a physical state-space realization, for direct output feedback with $H = G^T$, plus node 1 displacements and velocities

α^2 ($\times 10^{-5}$)	initial dv's (structural)	β^2	
		50	60
1	120.0	3516.5	3232.6
2	90.0	2402.4	2213.0
3	90.0	1935.5	1790.2
4	90.0	1671.5	1543.3
5	90.0	1492.9	1379.2
6	90.0	1364.6	1261.6
7	60.0	1459.5	1166.3
8	60.0	1184.0	1093.4
9	60.0	1116.8	1186.1
10	60.0	1063.4	983.3

Table 12: Optimal weight using continuation solution algorithm with a physical state-space realization, for direct output feedback with $H = G^T$ plus node 1 displacements and velocities

Lower case Greek letters represent scalars, upper case Roman represent matrices, while lower case Roman represent column vectors and indices. Superscripts T, H indicate transpose and conjugate transpose respectively. The notation k = i(r)j means that k takes all the values starting from i until j with step r. e_i represents the ith column of the identity matrix I, and $\lambda(\cdot)$ the set of eigenvalues of a matrix, counting multiplicities. $\mathbf{R}(\cdot)$, $\mathbf{N}(\cdot)$ will represent the space spanned by the columns of a matrix, and the null space of a matrix respectively. $\boldsymbol{R}, \mathbf{C}$ will represent the set of real and complex numbers respectively.

2 The Problem

The MEVAO problem that we will attack in this paper may be defined as follows:

We are given a controllable and observable system

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

$$y(t) = Cx(t) \tag{2}$$

where $A \in \boldsymbol{R}^{n \times n}$, $B \in \boldsymbol{R}^{n \times m}$, $C \in \boldsymbol{R}^{p \times n}$, rank (B)=m, rank(C)=p, $x(t) \in \boldsymbol{R}^n$ is the state of the system at time t and u(t) the input at time t. We are also given a self conjugate set of eigenvalues λ_i, $i = 1(1)r$ with r=min{n,m+p-1}, we need to compute $K \in \boldsymbol{R}^{m \times p}$ such that $\lambda(A - BKC) \supseteq \{\lambda_i \mid i = 1(1)r\}$. As a result, a linear output feedback $u(t) = -Ky(t)$ may be computed that will make (1) become

$$\dot{x}(t) = (A - BKC)x(t).$$

The resulting system will have desirable properties for the contror engineer.

3 Transmission Zeros

In the remaining of the paper unless otherwise stated, we will assume without loss of generality that $m \leq p$.

Transmission zeros ($trzs$) play an important role in the MEVAO, thus it is vital that they are well understood. Our experience with the literature on $trzs$ has been rather disappointing. In our search for a definition we have found several; some of them even contracting one another. Thus we

decided to resort to the physical motivation of *trzs*, and from that to derive a sensible definition. Some observations that may give some physical motivation to the concept of a transmission zero (of a controllable-observable system in our case) were found in [2], p.41. From this we concluded that, physically, a transmission zero (*trz*) of a system is a specific frequency of the system for which there exists a nonzero input that will yield a zero output. This basically is definition 1 below. In the following we will present three definitions of *trzs*, and we will prove their equivalence. The reason we present these three defintions is because the first is directly derived from the physical motivation of *trzs*, the second is very useful in matrix computations, and the third is frequently encountered in the literature.

Definition 1: A number $\zeta \in \mathbf{C}$ is a *trz* of (1),(2) if and only if there exists nonzero input u that can yield a zero output $y = G(\zeta)u = 0$, where

$$G(s) = C(sI - A)^{-1}B \quad , with \quad s \in \mathbf{C}$$

is the transfer function of (1),(2). □

The number of *trzs* of (1),(2) can be at most $n - max\{m, p\}$ (see for example [2], p.65). The set of all *trzs* of (1),(2) will be denoted by Z_{tr}. Often it is helpful to use the following definition of a *trz*.

Definition 2: $\zeta \in \mathbf{C}$ is a *trz* of (1),(2) if and only if

$$\exists u \neq 0 \ : \begin{pmatrix} A - \zeta I & B \\ C & 0 \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix} = 0 \quad with \quad x = (\zeta I - A)^{-1}Bu \ .$$

□

Frequently we encounter in the literature the following definition of *trzs*.

Definition 3: Let $L^{-1}(\zeta)G(\zeta)R^{-1}(\zeta) = \begin{pmatrix} \frac{\epsilon_1(\zeta)}{\psi_1(\zeta)} & & & \\ & \ddots & & \\ & & \frac{\epsilon_{m-1}(\zeta)}{\psi_{m-1}(\zeta)} & \\ & & & \frac{\epsilon_m(\zeta)}{\psi_m(\zeta)} \\ & & 0 & \end{pmatrix}$ be the Smith-McMillan (S-M) form of $G(\zeta)$, where $L(\zeta)$, $R(\zeta)$ are polynomial matrices and we have assumed that G(s) has normal rank m. Then ζ is a *trz* of (1) if and only if there exists $i = 1, \cdots, m$ such that $\epsilon_1(\zeta) = 0$. □

In the S-M form of $G(\zeta)$ the following properties hold

$$\epsilon_1(\zeta) \mid \epsilon_2(\zeta) \mid \cdots \mid \epsilon_m(\zeta) \quad \textit{and} \quad \psi_m(\zeta) \mid \psi_{m-1}(\zeta) \mid \cdots \mid \psi_1(\zeta) \, .$$

Because of the above properties we see that there cannot be $trz\ \zeta$ that will also be a root of $\psi_m(\zeta)$. Since, if it were then $\epsilon_m(\zeta) = 0$ and ζ would have been cancelled in $\epsilon_m(\zeta) \mid \psi_m(\zeta)$. Hence ζ would not be a trz, which is a contradiction. The roots of $\psi_i(\zeta)$, $i = 1(1)m$ are the poles of (1),(2), counting multiplicities. In our attempt to prove the equivalence of the above definitions we will be using the expression $G(\zeta)u$ with $\zeta \in Z_{tr}$. One then may wonder about the existance of $G(\zeta)u$ when ζ is also a pole of (1),(2). The following lemma will help us clarify this point.

Lemma 1: Let $\zeta \in \mathbf{C}$ be a trz as well as a pole of (1),(2), then there exists $u \neq 0$ such that $\lim_{s \to \zeta} G(s)u$ exists.

Proof: Let ζ be a pole as well as a trz of (1), then from the S-M form of $G(\zeta)$ we have that

$$\exists (i,j) \quad with \quad i < j \quad : \quad \psi_i(\zeta) = \epsilon_j(\zeta) = 0 \, .$$

Let us now for illustration assume $p = m = 3$ and $\psi_1(\zeta) = \epsilon_2(\zeta) = 0$. Then by taking $\tilde{u}^T = (0, \eta, \theta) \neq 0$ we have the following from the S-M form of G(s).

$$\begin{aligned} \lim_{s \to \zeta} L^{-1}(s)G(s)R^{-1}(s)\tilde{u} &= \begin{pmatrix} \lim_{s \to \zeta} \frac{\epsilon_1(s)}{\psi_1(s)} & & \\ & \lim_{s \to \zeta} \frac{\epsilon_2(s)}{\psi_2(s)} & \\ & & \lim_{s \to \zeta} \frac{\epsilon_3(s)}{\psi_3(s)} \end{pmatrix} \begin{pmatrix} 0 \\ \eta \\ \theta \end{pmatrix} \\ &= \begin{pmatrix} 0 \\ 0 \\ \theta \lim_{s \to \zeta} \frac{\epsilon_3(s)}{\psi_3(s)} \end{pmatrix} \end{aligned}$$

which is defined since $\psi_3(\zeta) \neq 0$. Since $L(s)$, $R(s)$ are nonsingular (they have determinants independent of s, see for example [7], p.21), $\lim_{s \to \zeta} L^{-1}(s)$, $\lim_{s \to \zeta} R^{-1}(s)$ are defined.

Hence if we take $u = \lim_{s \to \zeta} R^{-1}(s)\tilde{u}$ then $\lim_{s \to \zeta} G(s)u$ is defined for every $\zeta \in Z_{tr}$ that is also a pole of (1),(2). □

In the following when we write $G(\zeta)u$ with $\zeta \in Z_{tr}$ also being a pole of (1), we will mean $\lim_{s\to\zeta} G(s)u$. Similarly when we write $x = (\zeta I - A)^{-1}Bu$ we will mean $x = \lim_{s\to\zeta}(sI - A)^{-1}Bu$, when $\zeta \in Z_{tr}$ is also a pole of (1),(2).

Theorem 1: Definitions 1,2 and 3 are equivalent.

Proof: To prove that definition 1 is equivalent to definition 2 we proceed as follows:

$$\begin{aligned}
\zeta \in Z_{tr} &\overset{\text{def}_1}{\iff} \{\exists\, u \neq 0 \;:\; C(\zeta I - A)^{-1}Bu = 0\} \\
&\iff \{\exists\, u \neq 0 \;:\; Cx = 0 \quad with \quad x = (\zeta I - A)^{-1}Bu\} \\
&\iff \left\{\exists\, u \neq 0 \;:\; \begin{pmatrix} A - \zeta I & B \\ C & 0 \end{pmatrix}\begin{pmatrix} x \\ u \end{pmatrix} = 0 \quad with \quad x = (\zeta I - A)^{-1}Bu\right\} \\
&\overset{\text{def}_3}{\iff} \zeta \in Z_{tr}\ .
\end{aligned}$$

To prove the equivalence of definitions 1 and 3 we proceed as follows:

$$\begin{aligned}
\zeta \in Z_{tr} &\overset{\text{def}_1}{\iff} \{\exists\, u \neq 0 \;:\; G(\zeta)u = 0\} \\
&\iff \left\{\exists\, u \neq 0 \;:\; L(\zeta)\underbrace{\begin{pmatrix} \operatorname{diag}_{i=1(1)m} \frac{\epsilon_i(\zeta)}{\psi_i(\zeta)} \\ 0 \end{pmatrix}}_{M(\zeta)} R(\zeta)u = 0\right\} \qquad (3)
\end{aligned}$$

Since $L(s)$, $R(s)$ are nonsingular for $\forall s \in \mathbf{C}$, $M(\zeta)$ must not have full column rank in (2). Hence

$$\begin{aligned}
(3) &\iff \{\exists\, i \in \{1,2,\cdots,m\} \;:\; \epsilon_i(\zeta) = 0\} \\
&\overset{\text{def}_3}{\iff} \zeta \in Z_{tr}\ .
\end{aligned}$$

□

In the last theorem we silently assumed that the normal rank q of $G(s)$ is m, or equivalently the normal rank $n+q$ of $P(s) = \begin{pmatrix} A - sI & B \\ C & 0 \end{pmatrix}$ is $n+m$. However, theorem 1 holds even for the degenerate case where $q < min\{m,p\}$ $(= m$ according to our assumption). To show this we proceed as follows:

First we prove $Z_{tr} = \mathbf{C}$ using definition 1. Since $q < m$ the following is true

$$\{\forall \zeta \in \mathbf{C} \quad \exists u \neq 0 \quad : \quad G(\zeta)u = 0\} \overset{\text{def}_1}{\Longleftrightarrow} Z_{tr} = \mathbf{C}\,. \tag{4}$$

To prove the equivalence of definitions 1 and 2 we see

$$Z_{tr} = \mathbf{C} \overset{(4)}{\Longleftrightarrow} \left\{ \forall \zeta \in \mathbf{C} \quad \exists u \neq 0 \quad : \quad \begin{pmatrix} A - \zeta I & B \\ C & 0 \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix} = 0 \quad with \quad x = (\zeta I - A)^{-1} B u \right\}$$

To prove the equivalence of definitions 1 and 3 we see

$$\begin{aligned} Z_{tr} = \mathbf{C} \quad &\overset{(4)}{\Longleftrightarrow} \quad \left\{ \forall \zeta \in \mathbf{C} \;\; \exists u \neq 0 \;\; : \;\; L(\zeta) \begin{pmatrix} \underset{i=1(1)m}{\text{diag}} \frac{\epsilon_i(\zeta)}{\psi_i(\zeta)} \\ 0 \end{pmatrix} R(\zeta) u = 0 \right\} \\ &\Longleftrightarrow \quad \{\forall \zeta \in \mathbf{C} \;\; \exists i \in \{1, \cdots, m\} \;\; : \;\; \epsilon_i(\zeta) = 0\}. \end{aligned}$$

Hence another definition of *trzs* that is often found in literature, and according to which $\zeta \in Z_{tr}$ if and only if $G(\zeta)$ and $P(\zeta)$ go below their normal rank, does not agree in the degenerate case with definitions 1, 2 and 3. This is so because not $\forall \zeta \in \mathbf{C}$ will make $G(\zeta)$ and $P(\zeta)$ go below their normal rank, hence $Z_{tz} \neq \mathbf{C}$, according to this definition. In what follows we will assume that *trzs* are defined by definitions 1, 2 or 3.

Definition 4: A number $\zeta \in \mathbf{C}$ is a strong transmission zero (*strz*) of (1),(2) if and only if $G(\zeta) = 0$, or equivalently $\epsilon_1(\zeta) = 0$ (in the $S - M$ form of $G(\zeta)$), or equivalently

$$\left\{ \forall u \neq 0\,, \quad \begin{pmatrix} A - \zeta I & B \\ C & 0 \end{pmatrix} \begin{pmatrix} x \\ u \end{pmatrix} = 0 \;\;, \quad with \quad x = (\zeta I - A)^{-1} B u \right\}\,.$$

□

Theorem 2: $\zeta \in \mathbf{C}$ cannot be assigned by output feedback if and only if ζ is a strong transmission zero.

Proof: Let $G(\zeta) \neq 0$, then if $\zeta \in \lambda(A)$ we take $K = 0$ and ζ is placed. If $\zeta \notin \lambda(A)$ then there exists $u \neq 0$ such that

$$y = G(\zeta)u \neq 0 \;\; \Rightarrow \left\{ \begin{array}{l} x = (\zeta I - A)^{-1} B u \\ y = Cx \neq 0 \end{array} \right\}\,.$$

If we take $K = -\frac{uy^T}{y^T y}$ then we see that

$$\begin{aligned}[\zeta I - (A - BKC)]\, x &= (\zeta I - A)x + BKCx \\ &= Bu - B\frac{uy^T}{y^T y}\, y \;\; = 0\end{aligned}$$

Thus the eigenpair (ζ, x) has been placed, which proves the necessity of the propositionn given by the theorem. To prove its sufficiency assume $G(\zeta) = 0$, but nevertheless ζ has been placed. Then there exists $x \neq 0$ such that for some K

$$\begin{aligned}(A - \zeta I)x - BKCx = 0 \;\Rightarrow\; & x - (A - \zeta I)^{-1}BKCx = 0 \\ \Rightarrow\; & Cx - C(A - \zeta I)^{-1}BKCx = 0 \\ \Rightarrow\; & Cx + G(\zeta)KCx = 0 \\ \Rightarrow\; & Cx = 0\ .\end{aligned}$$

Hence from $(A - \zeta I)x - BKCx = 0 \;\Rightarrow\; (A - \zeta I)x = 0 \;\Rightarrow\; \zeta \in \lambda(A)$. This is a contradiction, since if $\zeta \in \lambda(A)$ the $G(\zeta)$ is not defined, however it is assumed that $G(\zeta) = 0$. □

4 The Algorithm

We give an algorithm based on deflation by unitary similarity transformations.

Step 1: Compute a feasible eigenvector x_1 of $A_1 - B_1K_1C_1 \equiv A - BKC$ corresponding to, say λ_1 with $\| x_1 \|_2 = 1$. We will show in the sequel how we may compute such an x_1.

Step 2: Allocate the eigenpair (λ_1, x_1) to $A_1 - B_1K_1C_1$ as follows: Let V_1 be a unitary transformation such that $V_1^H C_1 x_1 = \delta_1 e_1$ and $B_1 = (U_0, U_1)\begin{pmatrix} R_1 \\ 0 \end{pmatrix}$ a QR decomposition of B_1, then

$$\begin{aligned}(A_1 - B_1K_1C_1)x_1 = \lambda_1 x_1 \;\Longleftrightarrow\; & A_1x_1 - B_1(K_1V_1)(V_1^H C_1 x_1) \;=\; \lambda_1 x_1 \\ \Longleftrightarrow\; & B_1\tilde{K}_1\delta_1 e_1 \;=\; (A_1 - \lambda_1 I)x_1\ , \quad with \quad \tilde{K}_1 = K_1V_1 \\ \Longleftrightarrow\; & B_1 k_1 \delta_1 = (A_1 - \lambda_1 I)x_1\ , \quad with \quad k_1 = \tilde{k}_1 e_1 \\ \Longleftrightarrow\; & \begin{pmatrix} R_1 \\ 0 \end{pmatrix} k_1\delta_1 \;=\; \begin{pmatrix} U_0^H(A_1 - \lambda_1 I)x_1 \\ U_1^H(A_1 - \lambda_1 I)x_1 \end{pmatrix} \qquad (5)\end{aligned}$$

From (5) we see that if x_1 was computed in step 1 so that $x_1 \in \mathbf{N}[U_1^T(A_1 - \lambda_1 I)]$, and k_1 is computed by solving

$$R_1 k_1 = \delta_1^{-1} U_0^H (A_1 - \lambda_1 I) x_1 \tag{6}$$

then (5) is satisfied and the eigenpair (λ_1, x_1) is allocated. An eigenvector x_1 that satisfies $x_1 \in \mathbf{N}[U_1^T(A_1 - \lambda_1 I)]$ will be called feasible.

Step 3: Compute unitary $Q_1 = (x_1, \tilde{Q}_1)$. Hence from $Q_1 e_1 = x_1 \Rightarrow Q_1^H x_1 = e_1$ we see that Q_1 can be a Householder transformation or a sequence of rotations.

Step 4: Perform the unitary similarity transformation

$$\begin{aligned} Q_1^H(A_1 - B_1 K_1 C_1)Q_1 &= \begin{pmatrix} x_1^H \\ \tilde{Q}_1^H \end{pmatrix} [A_1 - B_1 \tilde{K}_1 (V_1^H C_1)](x_1, \tilde{Q}_1) \\ &= \begin{pmatrix} x_1^H \\ \tilde{Q}_1^H \end{pmatrix} [(A_1 x_1 - B_1 k_1 \delta_1)\,,\, (A_1 \tilde{Q}_1 - B_1 \tilde{K}_1 (V_1^H C_1)\tilde{Q}_1)] \end{aligned} \tag{7}$$

From step 3, k_1 has been computed so that $A_1 x_1 - B_1 k_1 \delta_1 = \lambda_1 x_1$. If we now set $\tilde{K}_1 = (k_1, K_2)$ and

$$\begin{aligned} V_1^H C_1 \tilde{Q}_1 &= \begin{pmatrix} c_1^H \\ C_2 \end{pmatrix} \quad \textit{then} \\ (7) &= \begin{pmatrix} x_1^H \\ \tilde{Q}_1^H \end{pmatrix} [\lambda_1 x_1 \;,\; A_1 \tilde{Q}_1 - B_1(k_1 c_1^H + K_2 C_2)] \\ &= \left(\begin{array}{c|c} \lambda_1 & x_1^H [A_1 \tilde{Q}_1 - B_1(k_1 c_1^H + K_2 C_2)] \\ \hline & \tilde{Q}_1^H A_1 \tilde{Q}_1 - \tilde{Q}_1^H B_1 k_1 c_1^H - \tilde{Q}_1^H B_1 K_2 C_2 \end{array}\right) \\ &= \left(\begin{array}{c|c} \lambda_1 & * \\ \hline & A_2 - B_2 K_2 C_2 \end{array}\right) \end{aligned}$$

where $A_2 = \tilde{Q}_1^H A_1 \tilde{Q}_1 - \tilde{Q}_1^H B_1 k_1 c_1^H$, $B_2 = \tilde{Q}_1^H B_2$.

Step 5: Continue the allocations with $A_2 - B_2 K_2 C_2$. □

There are two points that need to be clarified for the allocation of λ_1 in the above algorithm. One is, how to identify whether λ_1 is a *strz* or not, and the other is the computation of x_1.

Lemma 2: If $\{\lambda_1$ is a *strz* of (1) $\} \Rightarrow \delta_1 = 0$.

Proof: Let λ_1 be a *strz* of (1) then

$$\left\{\forall u_1 \neq 0 \ \exists x_1 \ : \ \begin{pmatrix} A_1 - \lambda_1 I & B_1 \\ C_1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ u_1 \end{pmatrix} = 0 \right\} \ \Rightarrow \ C_1 x_1 = 0 \ \Rightarrow \ \delta_1 = 0 \, .$$

We may note however that the above lemma on its own is not adequate to identify *strz*. In the sequel we will compute our feasible eigenvector x_1 in such a way that if $\delta_1 = 0$ then λ_1 will be a *strz*. As we will show, computing x_1 in this way will also improve the numerical stability of our algorithm. To accomplish the above we proceed as follows:

We know that

$$| \delta_1 | = \| C_1 x_1 \|_2 \Rightarrow \ \{\delta_1 = 0 \iff x_1 \in \mathbf{N}(C_1)\} \ .$$

Thus if we compute x_1 so that

$$x_1 \in \mathbf{N}[U_1^H (A_1 - \lambda_1 I)] - \mathbf{N}(C_1)$$

when

$$\mathbf{N}[U_1^H (A_1 - \lambda_1 I)] - \mathbf{N}(C_1) \neq \{\}$$

then we may safely conclude that $\delta_1 = 0 \ \Rightarrow \ \{\lambda_1$ is a *strz* $\}$. If the dimension of

$$\mathbf{N}[U_1^H (A_1 - \lambda_1 I)] - \mathbf{N}(C_1)$$

is greater than one, we have some freedom in choosing our x_1. To see how we may exploit this freedom, we observe from (6) that if $| \delta_1 |$ is unreasonably small then we may unnecessarily lose accuracy when we solve (6). To avoid this, let

$$C_1^T = (H_0, H_1) \begin{pmatrix} Z_1 \\ 0 \end{pmatrix}$$

be a QR decomposition of C_1^T, then $\mathbf{R}(H_0) = \mathbf{R}(C_1^T)$ and $\mathbf{R}(H_1) = \mathbf{N}(C_1)$. Assuming now $\| x_1 \|_2 = 1$ we get

$$| \delta_1 | = \| C_1 x_1 \|_2 = \| Z_1^T H_0^T x_1 \|_2 \leq \| Z_1 \|_2 \| H_0^T x_1 \|_2 \, .$$

But $\| H_0^T x_1 \|_2 = \sigma_{max}(H_0^T x_1) = cos\theta$, $\sigma_{max}(\cdot)$ is the maximum singular value of a matrix, and θ is the smallest angle between $\mathbf{R}(C_1^T)$ and $\mathbf{R}(x_1)$ (see[1]). Hence from

$$| \delta_1 | \leq \| z \|_2 \cos\theta$$

in order to make $| \delta_1 |$ as large as possible we need to make θ as small as possible. Thus we will choose $x_1 \in \mathbf{N}[U_1^H(A_1 - \lambda_1 I)]$ that forms an angle θ with $\mathbf{R}(H_0)$ as close to zero as possible. The following theorem taken from [1,p.582] and modified to meet our requirements, will give us a way to compute θ as it was required above.

Theorem 3: Let H_0 and W_1 form unitary bases of two subspaces. Let also

$$H_0^H W_1 = Y \Sigma X^H$$

be the singular value decomposition of $H_0^H W_1$, then the smallest angle between $\mathbf{R}(H_0$ and $\mathbf{R}(W_1)$ takes place between vectors $H_0 Y e_1$ and $W_1 X e_1$.

Proof: See [1]

Theorem 3 suggests the following algorithm for the computation of x_1:

Compute the QR decomposition of C_1^T, B_1, and $[U_1^H(A_1 - \lambda_1 I)]^H = (W_0, W_1) \begin{pmatrix} T \\ 0 \end{pmatrix}$.

Compute the singular value decomposition of $V_0^H W_1 = Y \Sigma X^H$.

Take $x_1 = W_1 X e_1$.

Finally to eliminate any doubt that $\delta_1 = 0$ when λ_1 is a *strz* we prove the following lemma.

Lemma 3: $\mathbf{N}[U_1^H(A_1 - \lambda_1 I)] \subseteq \mathbf{N}(C_1) \iff \{\lambda_1$ is a *strz* of (1),(2)$\}$

Proof: First we prove

$$\mathbf{N}[U_1^H(A_1 - \lambda_1 I)] = \{(\lambda_1 I - A_1)^{-1} B_1 u_1 \mid u_1 \in \mathbb{R}^m\} . \tag{8}$$

To do so, we see that, given $u_1 \neq 0$ and computing x_1 to satisfy

$$(A_1 - \lambda_1 I)x_1 = -B_1 u_1 \iff \begin{pmatrix} U_0^H \\ U_1^H \end{pmatrix} (A_1 - \lambda_1 I)x_1 = \begin{pmatrix} -R_1 \\ 0 \end{pmatrix} u_1$$
$$\Rightarrow \quad x_1 \in \mathbf{N}[U_1^H(A_1 - \lambda_1 I)].$$

Thus

$$\{\forall u_1 \in \mathbb{R}^m , (\lambda_1 I - A_1)^{-1} B_1 u_1 \in \mathbf{N}[U_1^H(A_1 - \lambda_1 I)]\} \Longrightarrow$$

$$\{(\lambda_1 I - A_1)^{-1} B_1 u_1 \mid u_1 \in \mathbb{R}^m\} \subset \mathbf{N}[U_1^H (A_1 - \lambda_1 I)] \,. \tag{9}$$

Let $x_1 : U_1^H (A_1 - \lambda_1 I) x_1 = 0$, then, since we can always compute

$$u \in \mathbb{R}^m : R_1 u_1 = U_0^H (A_1 - \lambda_1 I) x_1 \,,$$

we have $\begin{pmatrix} U_0^H \\ U_1^H \end{pmatrix} (A_1 - \lambda_1 I) x_1 = \begin{pmatrix} -R_1 \\ 0 \end{pmatrix} u_1 \Rightarrow x_1 = (\lambda_1 I - A_1)^{-1} B_1 u_1 \,,$

from which we get

$$\mathbf{N}[U_1^H (A_1 - \lambda_1 I)] \subset \{(\lambda_1 I - A_1)^{-1} B_1 u_1 \mid u_1 \in \mathbb{R}^m\} \,. \tag{10}$$

From (9), (10) we may derive (8).

Suppose now that

$$\begin{aligned} \{\lambda_1 \textit{ is a strz of } (1),(2)\} &\iff \{\forall u_1 \in \mathbb{R}^m \;\; C_1 (\lambda_1 I - A_1)^{-1} B_1 u_1 = 0\} \\ &\iff C_1 \mathbf{N}[U_1^H (A_1 - \lambda_1 I)] = 0 \\ &\iff \mathbf{N}[U_1^H (A_1 - \lambda_1 I)] \subset \mathbf{N}(C_1) \,. \end{aligned}$$

□

The algorithm that has been described so far in this section, can be used to allocate only one eigenvalue. We will show however that we may use it to allocate $min\{n, m+p-1\}$ eigenvalues. To do so we need to observe that only the first column of the current K_i is needed for the allocation of one eigenvalue. We also need to consider the fact that $\lambda(A_i - B_i K_i C_i) = \lambda(A_i^T - C_i^T K_i^T B_i^T)$. Given these two points then we may use the following algorithm, which for illustration we describe for $m = 2$ and $p = 3$, thus

$$K_1 = \begin{pmatrix} \times & \times & \times \\ \times & \times & \times \end{pmatrix} \,.$$

We allocate eigenvalues until the number of rows of the current K_i become greater than the number of columns. In our example this happens after the allocation of two eigenvalues, hence

$$K_3 = \begin{pmatrix} \times \\ \times \end{pmatrix} \,.$$

At this point we continue working with the transposed system $A_3^T - C_3^T K_3^T B_3^T$. Since K_3^T has two columns we are able to allocate two more eigenvalues instead of just one that we would have allocated if we had continued with K_3. In the general case we keep transposing until we run out of eigenvalues or columns. Thus the total number of eigenvalues that we manage to allocate using this algorithm is

$$m + p - 1 = 4 \ .$$

Note that by following this algorithm we also satisfy the condition $m \leq p$ at the beginning of each allocation. This condition has been assumed throughout this paper. Note that m, p are associated with the columns of the matrix on the left of K_i or K_i^T and the rows of the matrix on the right of K_i or K_i^T respectively, rather than with B_i and C_i.

5 Numerical Examples

In this section we give two numerical examples to demonstrate the performance of our algorithm. The computation was performed on double precision (56-bit mantissa) using PC-MATLAB on a Toshiba T5100 which uses an 80387 coprocessor equipped with the IEEE floating point standard of arithmetic. In the computation below we show, up to 12 decimal digits of accuracy.

Example 1:

$$A = \begin{pmatrix} 0.581314086914 & 0.504058837890 & 0.559921264648 & 0.741333007812 & 0.303421020507 \\ 0.166717529296 & 0.157394409179 & 0.665222167968 & 0.933609008789 & 0.144439697265 \\ 0.353500366210 & 0.441650390625 & 0.373367309570 & 0.771942138671 & 0.078048706054 \\ 0.836242675781 & 0.200820922851 & 0.072296142578 & 0.598373413085 & 0.638671875000 \\ 0.244094848632 & 0.936126708984 & 0.607955932617 & 0.154357910156 & 0.154678344726 \end{pmatrix}$$

$$B^T = \begin{pmatrix} 0.549163818359 & 0.843856811523 & 0.178649902343 & 0.409255981445 & 0.595489501953 \\ 0.942672729492 & 0.008666992187 & 0.239028930664 & 0.142791748046 & 0.671371459960 \\ 0.613098144531 & 0.065872192382 & 0.300445556640 & 0.576828002929 & 0.726318359375 \end{pmatrix}$$

$$C = \begin{pmatrix} 0.561660766601 & 0.332702636718 & 0.355514526367 & 0.279266357421 & 0.531448364257 \\ 0.427825927734 & 0.648269653320 & 0.666625976562 & 0.572494506835 & 0.972076416015 \\ 0.455917358398 & 0.134307861328 & 0.497573852539 & 0.212463378906 & 0.653732299804 \end{pmatrix}$$

The eigenvalues to be allocated are

$$\{0.704589843750, 0.421768188476, 0.572113037109, 0.396102905273, 0.127380371093\}.$$

The K computed by our algorithm was found to be

$$K = \begin{pmatrix} -9.415631257941 & -1.520241736469 & 11.520696978251 \\ -36.155174309311 & -4.515055559417 & 41.184483490595 \\ 33.590525809043 & 4.071903303950 & -36.923138190434 \end{pmatrix}$$

The eigenvalues of $A - BKC$ were computed and they were $\lambda(A - BKC) = \{0.704589843749, 0.421768188479, 0.572113037109, 0.396102905270, 0.127380371093\}$.

Example 2: In the above example we had $n = m + p - 1$, thus all n eigenvalues were allocated. However, if $m + p - 1 < n$ then $n - (m + p - 1)$ eigenvalues of $A - BKC$ will take values that our algorithm has absolutely no control over. The following example demonstrates this point.

$$A = \begin{pmatrix} 0.356336616284 & -0.356626507847 & -1.198870998736 \\ -0.210549376245 & 2.165165719183 & -0.882324378697 \\ -0.355939324751 & -0.506195941988 & 0.721098719251 \end{pmatrix}$$

$$B = \begin{pmatrix} 1.606955436561 & -0.407338058168 \\ 0.062823512419 & -0.595038063525 \\ -1.611627967397 & 0.616657720777 \end{pmatrix}$$

$$C = \begin{pmatrix} -1.182820557312 & 0.343687857622 & -0.357421120340 \end{pmatrix}$$

The eigenvalues to be allocated are {-0.406648486670, -0.366406484747, 0.853280016421}.

The K computed by our algorithm was found to be

$$K = \begin{pmatrix} -2.053475126112 \\ -6.466096811958 \end{pmatrix}$$

The eigenvalues of $A - BKC$ were computed and they were $\lambda(A - BKC)$ = {-0.406648486670, -0.366406484747, 1.707616832510}.

Our algorithm allocates one eigevalue at a time, thus complex eigenvalues need complex arithmetic. As a result, K may be complex. Investigation is under way to derive an algorithm that will allocate two eigenvalues at a time in a double step. In this way we will allocate a complex conjugate pair of eigenvalues in one double step using only real arithmetic. Hence K will be real.

6 Conclusion

We presented an algorithm for the pole assignment problem for multi-input, multi-output systems using output feedback. The algorithm uses deflation based on unitary similarity transformations and it allocates $min\{n, m + p - 1\}$ eigenvalues. The same kind of deflation has been used in [3] to solve the corresponding pole assignment problem using state feedback. Since the algorithm in [3] has been proven to be numerically stable we hope the algorithm in this paper has the same property too. However, this needs to be proven by doing a rounding error analysis of the algorithm, and we plan to do this in the near future.

References

[1] Bjorck, A. & Golub, G.H., "Numerical Methods for Computing Angles Between Linear Subspaces", Mathematics of Computation, v.27, No.123, pp. 579-594, 1973.

[2] Macfarlane, A.G.J. & Karcanias, N., "Poles and Zeros of Linear Multivariable Systems: A Survey of the Algebraic, Geometric and Complex-Variable Theory", Int. J. Control, v.24, No.1, pp. 33-74, 1976.

[3] Miminis, G.S. & Paige, C.C., "A Direct Algorithm for Pole Assignment of Time Invariant multi-input Linear Systems using State Feedback", Automatica, v.24, No.3, pp. 343-356, 1988.

[4] Miminis, G.S., "The use of Deflation in Eigenassignment Problems", To appear in the Proceedings of the MTNS89 conference, Amsterdam, June 1989.

[5] Parlett, B., "The Symmetric Eigenvalue Problem", Prentice-Hall, 1980.

[6] Saad, Y., "Projection and Deflation Methods for Partial Pole Assignment in Liner State Feedback", IEEE Trans. Autom. Control, v.33, No.3, pp. 290-297, 1988.

[7] Wilkinson, J., "The Algebraic Eigenvalue Problem", Oxford, 1965.