The Computation and Application of the Generalized Inverse via Maple

JON JONES \textsuperscript{1}, N. P. KARAMPETAKIS\textsuperscript{1} AND A. C. PUGH\textsuperscript{1,5}

\textsuperscript{1}Department of Mathematical Sciences, Loughborough University of Technology, England, U.K.
\textsuperscript{2}Department of Mathematics, Aristotle University of Thessaloniki, Thessaloniki 54006, Greece

In Karampetakis (1995) an algorithm for computing the generalized inverse of a singular polynomial matrix \( A(s) \in \mathbb{R}[s]^{n \times m} \) has been presented. In this paper the algorithm is extended to that of the singular rational matrix, \( A(s) \in \mathbb{R}(s)^{n \times m} \), and the algorithm is subsequently implemented in the symbolic computational package Maple. Several applications of its use are given.

© 1998 Academic Press Limited

1. Introduction

Consider a constant matrix \( A \in \mathbb{R}^{n \times m} \). For \( A \) non-singular (i.e. \( n = m \) and \( |A| \neq 0 \)) then it is known that there exists a unique non-singular matrix \( B \in \mathbb{R}^{m \times n} \) such that

\[
AB = BA = I_n
\]  

(1.1)

where \( I_n \) def the \((n \times n)\) identity matrix; here \( B \) is defined as the inverse of \( A \). However, for \( A \) singular (i.e. \( n \neq m \) or \( |A| = 0 \)) no such “two-sided” matrix inverse \( B \) exists. Clearly \( A \) possesses a “one-sided” inverse \( C \in \mathbb{R}^{m \times n} \) (resp. \( D \in \mathbb{R}^{m \times n} \)) such that

\[
AC = I_n \quad \text{(resp. } DA = I_m) \]

(1.2)

iff rank\([A]\) = \( n \) (resp. rank\([A]\) = \( m \)). Here \( C \) (resp. \( D \)) is termed a right (resp. left) inverse of \( A \).

Now consider the solution of a set of consistent linear equations of the form

\[
Ax = b
\]  

(1.3)

where \( A \in \mathbb{R}^{n \times m} \), \( x \in \mathbb{R}^{m} \) and \( b \in \mathbb{R}^{n} \). For \( A \) non-singular there exists a unique solution \( x = A^{-1}b \) where \( A^{-1} \) is the inverse of \( A \). For \( A \) singular, however, we have three distinct cases:

1. rank\([A]\) = \( n \) — a right inverse \( C \in \mathbb{R}^{m \times n} \) s.t. (1.2) holds. The vector \( x = Cb \) is a solution to (1.3) and is unique iff \( A \) is non-singular (i.e. \( C = A^{-1} \)).

\textsuperscript{5} E-mail: A.C.Pugh@lboro.ac.uk

0747-7171/98/010099 + 26 $25.00/0 sy970168 © 1998 Academic Press Limited
2. rank\(|A| = m\) a left inverse \(D \in \mathbb{R}^{m \times n}\) s.t. (1.2) holds. The vector \(x = Db\) is a solution to (1.3) if \(ADb = b\) and, if so, is unique in this case.

3. rank\(|A| = k \leq \min\{m, n\}\) — A solution of the form \(x = Pb\) may still exist.

Subsequently a more “generalized” class of inverse has been defined by Penrose (1955) which enables such solutions to be obtained. Indeed the problem of computing such generalized inverses has been considered by many authors (Ben-Israel and Greville, 1974, Decell, 1965; Fragulis et al., 1991; Karampetakis, 1995a; 1995c; Lovas-Nagy et al., 1978; Mertzios and Lewis, 1989; Penrose, 1955; Rao and Mitra, 1971). This is due to the large number of applications that such an inverse has in linear systems theory which include the calculation of the transfer function of a matrix (Fragulis et al., 1991), the solution of linear equations and the solution of diophantine equations (Karampetakis, 1995a).

In Decell (1965) an algorithm to compute a class of generalized inverse for a constant matrix \(A \in \mathbb{R}^{n \times m}\) has been given. This has recently been extended (Karampetakis, 1995a) for the more general singular polynomial matrix case

\[
A(s) \overset{\text{def}}{=} A_q s^q + \cdots + A_1 s + A_0 \in \mathbb{R}[s]^{n \times m}
\]

where \(A_i \in \mathbb{R}^{n \times m}, i = 0, \ldots, q\). In this paper we subsequently extend this algorithm to the singular rational matrix case \(A(s) \in \mathbb{R}(s)^{n \times m}\) and implement the resulting algorithm in the symbolic computational language Maple (Char et al., 1991).

In Section 2 we define several classes of generalized inverse and following Karampetakis (1995a) present an algorithm in Section 3 for the computation of a specific class of generalized inverse, defined as \(A(s)^t\), for a matrix \(A(s) \in \mathbb{R}(s)^{n \times m}\); the corresponding code for the computation of this inverse in Maple is also given. In Section 4 we consider several applications of this generalized inverse in linear systems theory and again the corresponding Maple code is presented for each such case. In Section 5 the implementation of these routines in Maple is discussed via the introduction of a linear systems package linsys. Finally in Section 6 these aforementioned Maple procedures are illustrated via several numerical examples and a critical appraisal of these results is presented.

2. Preliminaries

**Definition 2.1.** (A Generalized Inverse of \(A \in \mathbb{R}^{n \times m}\)) A generalized inverse of a matrix \(A \in \mathbb{R}^{n \times m}\) is defined as a matrix \(G \in \mathbb{R}^{m \times n}\) which satisfies at least the first or second of the following conditions:

\[
\begin{align*}
(i) \quad & AGA = A \\
(ii) \quad & GAG = G \\
(iii) \quad & [AG]^* = AG \\
(iv) \quad & [GA]^* = GA
\end{align*}
\]

(2.1)

where \([\cdot]^*\) denotes the conjugate transpose of the indicated function. For \(A\) non-singular then \(G \equiv A^{-1}\).

Analogously, following Karampetakis (1995a), we can define a generalized inverse of a polynomial matrix \(A(s) \in \mathbb{R}[s]^{n \times m}\) (resp. a rational matrix \(A(s) \in \mathbb{R}(s)^{n \times m}\)) as a matrix \(G(s) \in \mathbb{R}(s)^{m \times n}\) which satisfies at least the first or second equivalent polynomial (resp. rational) conditions in (2.1). Subsequently, from now on, we will consider the corresponding generalized inverse matrices, \(G(s) \in \mathbb{R}(s)^{m \times n}\), of a rational matrix only; clearly this is the most general case.
Various classes of generalized inverses $G(s)$ can be defined depending on which of conditions (2.1) they satisfy. It is usual to use the following notation (Ben-Israel and Greville, 1974).

**Definition 2.2. (The Classes of Generalized Inverse)**
Any matrix $A(s) \in \mathbb{R}(s)^{n \times m}$ which satisfies equations (a), (b), ..., from among (i)–(iv) in (2.1) will be termed an \{a,b,...\}-inverse of $A(s)$ and be denoted by $A(s)^{(a,b,...)}$.

Several classes of generalized inverse are defined below.

\{1\}-inverse $\overset{\text{def}}{=} A(s)^{(1)}$—Used to solve a set of consistent linear equations as in (1.3).

\{1,3\}-inverse $\overset{\text{def}}{=} A(s)^{(1,3)}$—Used to form a least squares solution to a set of inconsistent linear equations (1.3).

\{1,2,3\}-inverse $\overset{\text{def}}{=} A(s)^{(1,2,3)}$—Every right inverse of $A(s)$, as defined in (1.2), is a \{1,2,3\}-inverse of $A(s)$.

\{1,2,4\}-inverse $\overset{\text{def}}{=} A(s)^{(1,2,4)}$—Every left inverse of $A(s)$, as defined in (1.2), is a \{1,2,4\}-inverse of $A(s)$.

\{1,2,3,4\}-inverse $\overset{\text{def}}{=} A(s)^{\dagger}$—Used to form a minimum-norm least squares solution to a set of inconsistent linear equations (1.3).

The generalized inverse class $A(s)^{\dagger}$ is unique for each given matrix $A(s) \in \mathbb{R}(s)^{n \times m}$ and such a matrix always exists (Penrose, 1955). Further, it can be seen that

$$A(s)^{\dagger} \subset A(s)^{(1,2,3)} \subset \cdots \subset A(s)^{(1)}$$  \hspace{1cm} (2.2)

i.e. $A(s)^{\dagger}$ belongs to each and every generalized inverse class.

In the following paper we only consider the class $A(s)^{\dagger}$ and define this as "the generalized inverse" of $A(s)$.

3. **Computation of the Generalized Inverse via Maple**

In this section we present an algorithm for the computation of the generalized inverse, $A^{\dagger} \in \mathbb{R}(s)^{n \times m}$, of a singular rational matrix $A(s) \in \mathbb{R}(s)^{n \times m}$. The algorithm can be seen to be highly computationally attractive and the computational implications are subsequently discussed. The algorithm is subsequently implemented in the symbolic computational language Maple and the corresponding Maple code is presented.

3.1. **Algorithm**

The algorithm is a direct extension of Karampetakis (1995a) for the singular polynomial matrix case. Here we consider the variable $s$ to be an indeterminate although the same algorithm holds when $s \in \mathbb{C}$ under some slight modifications.
STEP 1. Consider $A(s) \in \mathbb{R}^{n \times n}$ and form the following sequences

$$
\begin{bmatrix}
A_0(s), A_1(s), \ldots, A_n(s) \\
a_0(s), a_1(s), \ldots, a_n(s) \\
B_0(s), B_1(s), \ldots, B_n(s)
\end{bmatrix}
$$

which are constructed recursively as follows:

$$
i = 0 \Rightarrow \begin{cases}
A_0(s) = 0_{n,n} \\
a_0(s) = 1 \\
B_0(s) = I_n
\end{cases}
$$

$$
i = 1, \ldots, n \Rightarrow \begin{cases}
A_i(s) = [A(s)A(s)^T] B_{i-1}(s) \\
a_i(s) = -\frac{\text{trace}(A_i(s))}{i} \\
B_i(s) = A_i(s) + a_i(s)I_n
\end{cases}
$$

STEP 2. Let $a_n(s) = \cdots = a_{k+1}(s) = 0$ while $a_k(s) \neq 0$. Then the Generalized Inverse, $A(s)^\dagger \in \mathbb{R}^{m \times n}$, of $A(s) \in \mathbb{R}^{n \times n}$ is given by

$$
A(s)^\dagger \overset{\text{def}}{=} -a_k(s)^{-1}A(s)^T B_{k-1}(s).
$$

If $k = 0$, (i.e. $a_n(s) = \cdots = a_1(s) = 0$), then

$$
A(s)^\dagger \overset{\text{def}}{=} 0_{m,n}
$$

the $(m \times n)$ zero matrix.

3.2. COMPUTATIONAL IMPLICATIONS

Several remarks concerning the algorithm and its computational implication can be made.

1. It is a simple recursion where each of the three sequences is added to at each step. Hence it is very computationally attractive.

2. The storage requirement can be significantly reduced via two means. First the sequence $[A_i(s), i = 0, \ldots, n]$, is not required in the final result and therefore we can update the sequence at each successive step. Secondly we only need to store the last non-zero $a_i(s)$ term. These savings would become more evident as the row order, $n$, of $A(s)$ is increased.

3. There is NO matrix inversion required and therefore the algorithm can be considered stable in this respect.

4. The dimension of the matrix sequences involved remain fixed throughout the algorithm and do not increase at any step.

5. The matrix $[A(s)A(s)^T] \in \mathbb{R}^{n \times n}$ in (3.2) is constant $\forall i \geq 1$ and subsequently only needs to be calculated once (i.e. outside the algorithm's main loop).

6. For $n > m$ we can form the transpose of $A(s)$, denoted by $A^T(s) \in \mathbb{R}^{m \times n}$, and compute the generalized inverse of this matrix instead. Then $A(s)^\dagger = [A^T(s)^\dagger]^T$. This will in general be computationally faster to compute as the algorithm will now be one of $m$ rather than $n$ steps.
7. The same algorithm holds when $A(s)$ is rational, polynomial or indeed constant; therefore only one computer procedure is required.

8. The algorithm holds for $n$-D matrices of the form $A(s_i), i = 1, \ldots, n$ (Karampetakis, 1965c).

### 3.3. Maple Code

The procedure to compute the generalized inverse $A(s)^\dagger \in \mathbb{R}(s)^{m \times n}$ of a rational matrix $A(s) \in \mathbb{R}(s)^{m \times m}$ is given below. The corresponding interface of the code with Maple will be discussed in Section 5.

```maple
# Procedure: GINVERSE(G)
# Parameters: 'G' - Matrix in a single indeterminant 's'
# Use: Computes the generalized inverse of a rational matrix in a
# single indeterminant
#
# Sub-Procedure: IDEN(x)
# Parameters: 'x' - Row/Column dimension
# Use: Forms the (x by x) identity matrix.

GINVERSE:=proc(G)
local n,m,NewG,Num,NewGtran,tag,a,k,Id,Gt,1,A;
#Define 'm'=row dimension of 'G'
#Define 'm'=column dimension of 'G'
    n:=rowdim(G);
    m:=coldim(G);
#Checks if array 'G' is of dimension (1 by 1)
#If YES then returns out of the procedure the trivial inverse
#If NO then continues through the procedure
    if n=1 and m=1 then
        if G[1,1]=0 then
            RETURN(G)
        else
            RETURN(inverse(G))
        fi;
    fi;
#Checks if 'n'>'m' due to the algorithm being 'm'-order dependent
#If YES then define
#- 'NewG'=transpose of 'G'
#- 'NewGtran'= 'G'
#If NO then define
#- 'NewG'='G'
#- 'NewGtran'=transpose of 'G'
    if n>m then
        NewG:=transpose(G);
        dum:=n;
        m:=dum;
        NewGtran:=copy(G);
        tag:=1
    else
        NewG:=copy(G);
        NewGtran:=transpose(G);
    fi;
end:
```

tag:=0
fi;

#define the following
#-\'a\'=1
#-\'ID\'=('n' by 'n') identity matrix
#-\'B0\'=('n' by 'n') identity matrix
#-\'Gt\'='NewG' x 'NewGtran'
a:=i;
k:=0;
ID:=IDEN(n);
B,k:=copy(ID);
Gt:=evalm(NewG &* NewGtran);

#Main algorithm component: Consists of 'n'-steps with counter variable 'i'.
#For each step we apply the following
#- Check if the trace of 'Gt' x 'B.(i-1)' is non-zero
#- If YES then update 'a' which is non-zero, 'B.i' and variable 'k'=i'. If NO then form 'B.i' only
#- We leave loop with the following
#- 'k'-highest value of 'i' for which 'a' is non-zero
#- 'a'=-trace('Gt' x 'B.(k-1)')/k
#- the sequence 'B0', ..., 'Bn'
for i from 1 to n do
  if i=1 then
    if trace(Gt)>0 then
      a:=trace(Gt);
      B.i:=evalm(Gt+a*ID);
      k:=1
    else
      B.i:=evalm(Gt)
    fi
  else
    A:=evalm(Gt &* B.(i-1));
    if trace(A)>0 then
      a:=trace(A)/i;
      B.i:=evalm(A+a*ID);
      k:=i
    else
      B.i:=evalm(A)
    fi
  fi
od;

#Output the generalized inverse
#If 'k'=0 then return the zero matrix
#If 'k'=0 then compute via the given formulae depending on value of 'tag'
if tag=1 then
  if k=0 then
    matrix(n,m,0)
  else
    map(factor,map(normal,transpose(evalm(-(1/a*NewGtran &*\ B.(k-1))))))
  fi
else
  if k=0 then
    matrix(n,m,0)
  else
    map(factor,map(normal,evalm(-(1/a*NewGtran &* B.(k-1)))))
4. Applications of the Generalized Inverse

In this section we present three applications for the generalized inverse, $A^\dagger \in \mathbb{R}(s)^{m \times n}$, of a singular rational matrix $A(s) \in \mathbb{R}(s)^{n \times m}$ in linear systems theory. For each of these applications the corresponding Maple code is given.

4.1. Solution of $A(s)X(s)B(s) = C(s)$

Consider the solution $X(s) \in \mathbb{R}(s)^{m \times k}$ of the matrix equation

$$A(s)X(s)B(s) = C(s) \quad (4.1)$$

where $A(s) \in \mathbb{R}(s)^{n \times m}$, $B(s) \in \mathbb{R}(s)^{k \times l}$ and $C(s) \in \mathbb{R}(s)^{m \times l}$. From Lovass-Nagy et al. (1978) we have the following

**Theorem 4.1. (The Solution of $A(s)X(s)B(s) = C(s)$)**

The equation $A(s)X(s)B(s) = C(s)$, where $A(s) \in \mathbb{R}(s)^{n \times m}$, $B(s) \in \mathbb{R}(s)^{k \times l}$ and $C(s) \in \mathbb{R}(s)^{m \times l}$, has a solution $X(s) \in \mathbb{R}(s)^{m \times k}$ if and only if the consistency condition

$$A(s)A(s)^\dagger C(s)B(s)^\dagger B(s) = C(s) \quad (4.2a)$$

is satisfied in which case all the solutions are given by

$$X(s) = A(s)^\dagger C(s)B(s)^\dagger + Y(s) - A(s)^\dagger A(s)Y(s)B(s)B(s)^\dagger \quad (4.2b)$$

where $A(s)^\dagger$ and $B(s)^\dagger$ are the generalized inverses of $A(s)$ and $B(s)$ respectively and $Y(s)$ is arbitrary to within having the same dimension as $X(s)$.

The above theorem similarly holds when $A(s)^\dagger$, $B(s)^\dagger$ are replaced with specific $\{1\}$-inverses, $A^{(1)}$, $B^{(1)}$, respectively.

We can apply the above theorem to several areas. Clearly for the case $B(s) = C(s) = A(s)$, $(4.2b)$ gives all possible $\{1\}$-inverses of $A(s)$ given a single $\{1\}$-inverse; we can form other classes of generalized inverses in an analogous manner. Other applications include the solution of AR-Representations (Karampetakis, 1995b), feedback compensation and matching problems (Karampetakis, 1995c) and the solution of diophantine equations; all of these will be considered in this paper.
4.1.1. MAPLE CODE

The procedure to compute the solution of the matrix equation (4.1) is given below. The corresponding interface of the code with Maple will be discussed in Section 5.

```maple
# Procedure: PLINSOLVE(A,B,C)
# Parameters: 'A', 'B', 'C' - Matrices in equation (4.1)
# Use: Solves the matrix equation A(s)X(s)B(s)=C(s) as in (4.1)
# if a consistent solution exists

PLINSOLVE:=proc(A,B,C)
local GA, m, GB, k, Y, Cn, left;

# Define 'GA'=generalized inverses of 'A'
GA:=GINVERSE(A);
m:=rowdim(GA);

# Define 'GB'=generalized inverses of 'B'
GB:=GINVERSE(B);
k:=coldim(GB);

# Test the consistency condition for a solution
# Define 'Cn'=simplified form of 'C'
# Define 'left'=left hand side of consistency equation (4.2a)
Cn:=map(normal,C);
left:=map(normal,evalm(A &* GA &* Cn &* GB &* B));

# Test if 'left'=Cn
# If YES define 'Y'=('m' by 'k') matrix and compute solution
if equal(left,Cn)=true then
    Y:=matrix(m,k);
    map(normal,evalm(GA &* Cn &* GB + Y - GA &* A &* Y &* B &* GB));
else
    RETURN('the system is not solvable');
end;
end;
```

4.2. SOLUTION OF AUTOREGRESSIVE(AR)-REPRESENTATIONS

Consider the AR-Representation

\[
A(\rho)\beta(t) = 0
\]

where \( \beta(t) : [0-, +\infty) \rightarrow \mathbb{R}^m \), \( \rho \stackrel{\text{def}}{=} \frac{d}{dt} \) denotes the differential operator and

\[
A(\rho) \stackrel{\text{def}}{=} A_0\rho^0 + \cdots + A_1\rho + A_0 \in \mathbb{R}^{n \times m}
\]

where \( A_i \in \mathbb{R}^{n \times m}, i = 0, 1, \ldots, q \) and \( n \) is not necessarily equal to \( m \). Using Laplace transforms (Pugh, 1976) the form

\[
A(s)\tilde{\beta}(s) = \begin{pmatrix} s^{q-1}I_n & s^{q-2}I_n & \cdots & I_n \\ S_{q-1} & \end{pmatrix} \begin{pmatrix} A_0 & 0 & \cdots & \cdots & \cdots & \cdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ A_1 & \cdots & A_q \\ X_A & \end{pmatrix} \begin{pmatrix} \beta(0-) \\ \vdots \\ \beta(q-1)(0-) \\ \beta(0-) \end{pmatrix} \stackrel{\text{def}}{=} \tilde{a}(s)
\]

(4.3)
where $L[\beta(t)] = \hat{\beta}(s)$, the Laplace transform of $\beta(t)$. From Theorem 4.1 we have the following result:

**Theorem 4.2.** (Solution of $A(\rho)\beta(t) = 0$) The AR-Representation (4.3) has a solution iff

$$A(s)A(s)^\dagger \hat{\alpha}(s) = \hat{\alpha}(s)$$

(4.6a)

in which case all general solutions are given by

$$\beta(t) = L^{-1}[\beta(s)] = L^{-1}[A(s)^\dagger \hat{\alpha}(s) + [I_m - A(s)^\dagger A(s)]y(s)]$$

(4.6b)

where $y(s)$ is arbitrary to within having the same dimension as $\hat{\beta}(s)$ and $L^{-1}[\cdot]$ denotes the inverse Laplace transform of the indicated function.

### 4.2.1. Maple Code

The procedure to compute the solution of the AR-Representation (4.3) is given below. The corresponding inclusion of the code in Maple will be discussed in Section 5.

```
# Procedure: ARSOLVE(A,B,q)
# Parameters: 'A' - the matrix A(s) in (4.3)
# 'B' - initial condition vector beta(0-) in (4.5)
# 'q' - integer value of degree of A(s)
#
# Use: Solve the AR-Representation A(rho)beta(t)=0 as in (4.3)
# if a consistent solution exists

ARSOLVE:=proc(A,B,q)
local n,m,ExpA,b,c,d,CD,termA,e,termB,i,j,ys,asht,C,Left,Xs;
n:=rowdim(A);#
m:=coldim(A);
#Define 'ExpA'=expanded form of 'A'
ExpA:=map(expand,A);
#Compute the matrix sequence 'A0', 'A1', ..., 'Aq' as in (4.4)
#Define 'b'=main counter variable
#For each step we apply the following
#-Define 'A.b'=initial zero ('n' by 'm') matrix
#-Copy each element of 'ExpA' of degree s'b to 'A.b'
for b from q by -1 to 0 do
    A.b:=matrix(n,m,0);
    for c from 1 to m do
        for d from 1 to n do
            CO:=coeff(ExpA[c,d],s,b);
            A.b[c,d]:=CO
        od
    od;
#Compute the augmented matrix S_(q-1) as in (4.5)
#Define 'termA'='n' by 'n' identity matrix; this will represent S_(q-1)
    termA:=Identity(n);
#Define 'e'=counter variable
    for e from 1 to q-1 do
        termA:=augment(s*e + Identity(n),termA)
    od
end;
```
od;
# Compute the block matrix $X_{-}(A)$ as in (4.5)
# Define 'terms' = ('q*m' by 'q*n') zero matrix; this will represent $X_{-}(A)$
termB:=matrix(q=m, q=n, 0);
# Copy the matrices 'A0', ..., 'Aq' into this matrix
for i from 1 to q do
  for j from 1 to q do
    if i >= j then
      termB:=copyinto(A,(q-(i-j)),termB,(i-1)*n+1,(j-1)*n+1)
    fi
  od;
# Compute the $a^{-1}(s)$ vector as in (4.5)
# Define 'ashat'=left hand side of (4.5); this will represent $a^{-1}(s)$
ashat:=map(normal, evalm(termA &* termB &* B));
# Define 'GA'=generalized inverse of 'A'
GA:=GINVERSE(A);
# Test the consistency condition for a solution
# Define 'left'=left hand side of consistency equation (4.6a)
left:=map(normal, evalm(A &* GA &* ashat));
# Test if 'left'=ashat'
# If YES define 'ys'=('m' by '1') vector and compute solution
# If NO return no solution
if equal(left, ashat) then
  ys:=vector(n);
  Xs:=map(normal, evalm(GA &* ashat + evalm(normal(IDEN(n) - GA &* A) &* ys)));
  map(invlaplace,Xs,s,t)
else
  RETURN('the system is not solvable')
fi
end;

4.3. FEEDBACK PROBLEMS

Consider an open loop system with transfer function matrix $G(s) \in \mathbb{R}(s)^{n \times m}$ as shown in Figure 1 where $u(s) \in \mathbb{R}(s)^{n}$ and $y(s) \in \mathbb{R}(s)^{m}$ respectively represent the input and output to the system; alternatively $y(s) = G(s)u(s)$.

![Figure 1. The open-loop system.](image)

We are interested in two distinct problems namely those of "Feedback Compensation" and "Feedback Matching".

i Feedback Compensation

Here our aim is to find whether there exists an output feedback of the form

$$u(s) = -F(s)y(s) + v(s), \quad (4.7)$$
where \( F(s) \in \mathbb{R}(s)^{m \times n} \), such that the resulting closed-loop system, as shown in Figure 2, has a specified transfer function matrix \( H(s) \in \mathbb{R}(s)^{n \times m} \); alternatively \( y(s) = H(s)v(s) \).

Secondly if such an output feedback exists it is required to compute the controller \( F(s) \). It is clear that for this to occur the following condition must be satisfied

\[
H(s) = (I_n + G(s)F(s))^{-1}G(s) \Leftrightarrow G(s)F(s)H(s) = G(s) - H(s) \tag{4.8}
\]

where it is assumed that \((I_n + G(s)F(s))\) is invertible.

Again from Theorem 4.1 we have the following:

**Theorem 4.3. (Feedback Compensation Solution)** (4.8) has a solution iff

\[
G(s)[G(s) - H(s)\overline{H(s)}]H(s) = G(s) - H(s) \tag{4.9a}
\]

in which case all feedback compensators \( F(s) \) are given by

\[
F(s) = G(s)^\dagger[G(s) - H(s)]H(s)^\dagger + Y(s) - G(s)\overline{Y(s)H(s)\overline{H(s)}} \tag{4.9b}
\]

where \( Y(s) \) is arbitrary to within having the same dimension as \( F(s) \).

**ii Feedback Matching**

Here our aim is to find whether there exist compensator matrices \( F(s) \in \mathbb{R}(s)^{m \times n} \) and \( K(s) \in \mathbb{R}(s)^{m \times m} \) such that the resulting closed-loop system, as shown in Figure 3, has a specified transfer function matrix \( H(s) \in \mathbb{R}(s)^{n \times m} \); alternatively \( y(s) = H(s)v(s) \).

Secondly if such a compensator pair \([F(s) \ K(s)]\) exist it is required to compute them. It is clear that for such a pair to exist the following condition must be satisfied:

\[
H(s) = (I_n + G(s)F(s))^{-1}G(s)K(s) \Leftrightarrow G(s)F(s)H(s) = G(s)K(s) - H(s) \tag{4.10}
\]

where it is assumed that \((I_n + G(s)F(s))\) is invertible; this can be rewritten in the matrix form

\[
G(s)[F(s) \ K(s)]\begin{bmatrix}
H(s) \\
-I_m
\end{bmatrix} = -H(s). \tag{4.11}
\]

Again from Theorem 4.1 we have the following:
Theorem 4.4. (Feedback Matching Solution) (4.11) has a solution iff

\[-G(s)G(s)^T H(s) \begin{bmatrix} H(s) \\ -I_m \end{bmatrix}^\dagger \begin{bmatrix} H(s) \\ -I_m \end{bmatrix} = -H(s)\]  

(4.12a)

in which case all pairs \([F(s) \ K(s)]\) are given by

\[|F(s)K(s)| = -G(s)^TH(s) \begin{bmatrix} H(s) \\ -I_m \end{bmatrix}^\dagger + Y(s) - G(s)^TG(s)Y(s) \begin{bmatrix} H(s) \\ -I_m \end{bmatrix} \begin{bmatrix} H(s) \\ -I_m \end{bmatrix}^\dagger\]  

(4.12b)

where \(Y(s)\) is arbitrary to within having the same dimension as the augmented matrix \([F(s) \ K(s)]\).

From the above two theorems we respectively obtain the entire set of solutions for each problem. We may therefore select parameter values inherent in each such solution to obtain, for example, a stable system, a proper compensator \(F(s)\) or proper compensator pair \([F(s) \ K(s)]\) or whatever.

4.3.1. Maple code

The procedures to solve the Feedback Compensation and Feedback Matching problems are given below. The corresponding interface of the code with Maple will be discussed in Section 5.

i Feedback Compensation

```
# Feedback Compensation

# Procedure: CONSOLVE(G,H)
# Parameters: 'G' - open loop transfer function matrix
#             'H' - desired closed loop transfer function matrix
#
# Use: To compute an output feedback F(s) to give a desired closed loop transfer function matrix H(s)

CONSOLVE:=proc(G,H)
```


PLINSOLVE(G,H,evalm(G - H))
end;

ii Feedback Matching

MATCH:=proc(G,H,Fa,Ka)
local n,m,HI,FK;
   n:=rowdim(G);
   m:=coldim(G);
#Define 'HI' as stacked matrix in (4.11)
   HI:=stack(H,evalm(-1 * IDEN(m)));
#Define 'FK'=the augmented solution matrix \([F(s) \ K(s)]\)
   FK:=PLINSOLVE(G,HI,evalm(-1 * H));
#Output solution matrices
   print('F is given by',map(normal,submatrix(FK,1..m,1..n)));
   print('K is given by',map(normal,submatrix(FK,1..m,n+1..m+n)));
#If number of arguments>2 then assign 'Fa' and 'Ka' to the solution
#matrices
   if nargs>2 then
      Fa:=map(normal,submatrix(FK,1..m,1..n));
      Ka:=map(normal,submatrix(FK,1..m,n+1..m+n))
   fi
end;

5. Implementation via Maple

In this section we interface the procedures implemented above with the symbolic computational package Maple (Char et al., 1991) via the introduction of a linear systems package linsys. This is preceded by a brief overview of the Maple software.

5.1. The Symbolic Package Maple

Although symbolic packages can be regarded as being in general “slow” their advantages, particularly in the case of Maple, lie in the following:

Symbolic—Variables can be stored in “exact” form (i.e. 1/3 as opposed to 0.3333...) resulting in no loss of accuracy during a calculation. Additionally variables can be left “unassigned” (i.e. without holding any numerical values) which enables polynomial operations to be defined, say, in the indeterminate \(x\).

Procedures—Inbuilt procedures exist which the user can implement directly, either
individually or in unison. In Maple the linalg package consists of a collection of procedures for matrix manipulation in linear algebra.

**Programming**—The ability to write new procedures using inbuilt high-level programming languages which can utilize any other existing procedure. In this respect the inbuilt procedures within Maple can be viewed upon as building blocks for more specialized and advanced procedures.

5.2. THE **linsys** PACKAGE

Maple enables new packages, or groups of related procedures, to be formed and implemented directly. The advantage of forming packages is that they are easier to work with and groups of procedures can be read into a worksheet as a whole instead of individually. A package is read into a Maple session via the command

\[ \text{with(package):} \]

A procedure, say file1, existing within the package can be subsequently implemented via

\[ \text{file1(args1);} \]

The **linsys** package is such a package which is concerned with the solution of linear systems. The package is divided into three separate sub-packages

**linstruct**—This package contains 20 procedures concerned with determining the structure and properties of a system; it can be seen to be a natural extension to the linalg package inherent within Maple. Such procedures include those for computing row and column proper matrices; the GINVERSE procedure is included here.

**linrep**—This package contains 15 procedures used to determine alternative and equivalent representations of a system. Such procedures include those for computing Matrix Fraction Descriptions.

**linsol**—This package contains 20 procedures concerned with obtaining solutions to a system; it is primarily concerned with computing the admissible solutions to ARMA-Representations. The PLINSOLVE, COMSOLVE and MATCH procedures are included here.

6. Examples

In this section we use the procedures, as presented in Sections 3 and 4, via Maple; the machine used is a SUN SPARC station10 (75MHz SuperSPARC II). The intermediate output times, initiated via the **showtime()** procedure, indicate the CPU time used (in seconds) in the computation.

6.1. THE GENERALIZED INVERSE OF A SINGULAR MATRIX

Consider the singular matrix \( B(s) \in \mathbb{R}^{2 \times 3} \)

\[
B(s) = \begin{pmatrix}
1 & s + 1 & s \\
1 & 3 & 1 \\
&s & s + 1 & s + 2
\end{pmatrix}.
\]  
(6.1)
The normal inverse of $B(s)$ is clearly not defined; however, the generalized inverse of $B(s)$ can be computed via the implementation of the procedure GINVERSE. This is illustrated below.

> with(linalg):
Warning: new definition for norm 
Warning: new definition for trace 

> with(linsys):

> B:=matrix(2,3,[1,s+1,s,1/(s+1),1/(s+2)]);

> B :=
[ 1 s + 1 s ]
[ ]
[ 3 1 ]
[ 1/(s+1) ---- ---- ]
[ ]
[ s + 1 s + 2 ]

> readlib(showtime): showtime();

01 :=

> inverse(B);
Error, (in inverse) expecting a square matrix

time 0.01 words 3252
02 :=

> GINVERSE(B);

\[
\begin{align*}
3 & 2 & 4 \\
\text{s (12 s + 6 s + s + 12 + 4 s)} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\text{\%1} & & \\
\end{align*}
\]

\[
\begin{align*}
\text{s (s + 1) (s + 2) (- s + 2 + 2 s + 2 s)} & & 2 & 3 \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\text{\%1} & & & & \\
\end{align*}
\]

\[
\begin{align*}
\text{s (s + 1) (s + 2) (3 s - 2 s + 2 s + s - 2)} & & 3 & 2 & 4 \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\end{align*}
\]

\[
\begin{align*}
\text{s (7 s + 29 s + 30 s + s + 2)} & & 3 & 2 & 4 \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\text{\%1} & & & & & & \\
\end{align*}
\]

\[
\begin{align*}
\text{\%1} & & 17 s - 2 s + 7 s - 4 s + 4 + 6 s + 22 s
\end{align*}
\]
It can subsequently be shown that the generalized inverse matrix computed above satisfies the four properties of Definition 2.1.

6.2. THE SOLUTION OF $A(s)W(s) + B(s)Y(s) = C(s)$

Consider the polynomial matrix diophantine equation

$$
\begin{bmatrix}
1 & 0 \\
0 & s
\end{bmatrix} W(s) + \begin{bmatrix}
1 \\
0
\end{bmatrix} Y(s) = \begin{bmatrix}
s & 1 \\
s & 0
\end{bmatrix}
$$

(6.2)

where $A(s) \in \mathbb{R}[s]^{2 \times 2}$, $B(s) \in \mathbb{R}[s]^{2 \times 1}$ and $C(s) \in \mathbb{R}[s]^{2 \times 2}$; we are therefore interested in a polynomial solution pair $W(s) \in \mathbb{R}[s]^{2 \times 2}$ and $Y(s) \in \mathbb{R}[s]^{1 \times 2}$. We can write (15) in the form

$$
\begin{bmatrix}
1 & 0 & 1 \\
0 & s & 0
\end{bmatrix}
\begin{bmatrix}
W(s) \\
Y(s)
\end{bmatrix}
= \begin{bmatrix}
s & 1 \\
s & 0
\end{bmatrix}
$$

(6.3)

Clearly (6.3) is of the form (4.1) where $A(s) \in \mathbb{R}[s]^{2 \times 3}$, $B(s) = I_2$ and $C(s) \in \mathbb{R}[s]^{2 \times 2}$ and we require to solve (6.3) for a polynomial solution $X(s) \in \mathbb{R}[s]^{3 \times 2}$. It can be seen that the reduced consistency condition (4.2a) is satisfied and therefore a solution to (6.3) exists. This solution will be polynomial as every left divisor of $A(s)$ in (6.3) is also a left divisor of $C(s)$.

We have two methods to solve (6.3) via Maple:

1. **linsolve**—This is an inherent procedure contained within the *linalg* package of Maple which solves the consistent linear matrix equation $A(s)X(s) = C(s)$.
2. **PLINSOLVE**—This is a developed procedure contained within the *linsol* package which solves the more general consistent linear matrix equation $A(s)X(s)B(s) = C(s)$ via the computation of the generalized inverse $A(s)^{\dagger}$ of $A(s)$. Although only a $\{1\}$-inverse is required to solve this equation the generalized inverse $A(s)^{\dagger}$ is applicable to a wider class of problems and is therefore preferred.

The linsolve procedure can in general be seen to be the faster of the two procedures. However, the computed solution, $X(s) \in \mathbb{R}[s]^{m \times k}$, is generally under-parametrized for $k > 1$. Given $A(s)X(s) = C(s)$, where $A(s) \in \mathbb{R}[s]^{m \times m}$ (rank($A(s)$) = $r$), $X(s) \in \mathbb{R}[s]^{m \times k}$ and $C(s) \in \mathbb{R}[s]^{n \times k}$, we expect $k(m - r)$ independent parameters in the solution, $(m - r)$ parameters connected with each column of $X(s)$. Although the linsolve procedure only realizes $(m - r)$ parameters in total, these $(m - r)$ parameters are seen to be repeated in each column of the solution. This, although sufficient, is clearly not necessary and therefore the solution can be considered to be only "trivially" under-parametrized. The PLINSOLVE procedure, when applied to the equation $A(s)X(s) = C(s)$, is generally over-parametrized i.e. it contains more than $k(m - r)$ parameters. We can, however, reduce the number of dependent parameters in this case to form an independent set.
In these cases the solutions obtained via the procedures linsolve and PLINSOLVE can be seen to be equivalent. The implementation of both these procedures via Maple is given below.

\begin{verbatim}
> with(linalg):
Warning: new definition for norm
Warning: new definition for trace

> with(linsys):

> AB := matrix(2, 3, [1, 0, 1, 0, 0, 0, 0, 0, 0]);

\begin{bmatrix}
1 & 0 & 1 \\
0 & s & 0
\end{bmatrix}

> C := matrix(2, 2, [s, 1, s, 0]);

\begin{bmatrix}
s & 1 \\
0 & s
\end{bmatrix}

> readlib(showtime): showtime();

01 :=

> sol1 := PLINSOLVE(AB, IDEN(2), C);

\begin{bmatrix}
\frac{s}{2} + \frac{1}{2} Y[1, 1] - \frac{1}{2} Y[3, 1] & \frac{1}{2} + \frac{1}{2} Y[1, 2] - \frac{1}{2} Y[3, 2] \\
1 & 0
\end{bmatrix}

PLIN :=

\begin{bmatrix}
\frac{s}{2} + \frac{1}{2} Y[3, 1] - \frac{1}{2} Y[1, 1] & \frac{1}{2} + \frac{1}{2} Y[3, 2] - \frac{1}{2} Y[1, 2]
\end{bmatrix}

time 0.86 words 122368

02 :=

> sol2 := linsolve(AB, C);

\begin{bmatrix}
s - t[1] & 1 - t[1] \\
1 & 0
\end{bmatrix}

LIN :=

\begin{bmatrix}
-t[1] & -t[1]
\end{bmatrix}

time 0.26 words 8601

03 :=

> off
\end{verbatim}

We expect the solution \( X(s) \in \mathbb{R}[s]^{3 \times 2} \) to contain \( k(m-r) = 2(3-2) = 2 \) independent parameters. From the above, the PLINSOLVE output, sol1, contains four parameters \( Y_{1,1}, Y_{3,1}, Y_{1,2}, Y_{3,2} \) (i.e. it is over-parametrized) while the linsolve output, sol2, contains only one parameter \( t_1 \) (i.e. it is under-parametrized).
For the substitution

\[ p = Y_{1,1} - Y_{3,1}, \quad q = Y_{1,2} - Y_{3,2} \]  \hspace{1cm} (6.4)

**sol1** takes the form

\[
\text{sol1} = \begin{bmatrix}
\frac{1}{2} (s + p) & \frac{1}{2} (1 + q) \\
1 & 0 \\
\frac{1}{2} (s - p) & \frac{1}{2} (1 - q)
\end{bmatrix}
\]  \hspace{1cm} (6.5)

which contains the two independent parameters \( p \) and \( q \) as required.

In **sol2** the parameter \( \mathcal{J}_1 \) is repeated in both of its columns. We can therefore form a complete solution by replacing \( \mathcal{J}_1 \) in the second column by a second independent parameter, \( \mathcal{J}_2 \), resulting in

\[
\text{sol2} = \begin{bmatrix}
s - \mathcal{J}_1 & 1 - \mathcal{J}_2 \\
1 & 0 \\
\mathcal{J}_1 & \mathcal{J}_2
\end{bmatrix}
\]  \hspace{1cm} (6.6)

The two solutions (6.5) and (6.6) can be seen to be identical under the parameter relation

\[ \mathcal{J}_1 = \frac{1}{2} (s - p), \quad \mathcal{J}_2 = \frac{1}{2} (1 - q). \]  \hspace{1cm} (6.7)

### 6.3. Solution of AR-Representations

Consider the AR-Representation

\[
\begin{bmatrix}
\rho & \rho^3 & \rho^2 + \rho \\
1 & \rho^3 & \rho + 1 \\
0 & \rho + 1 & 0
\end{bmatrix}
\begin{bmatrix}
\beta_1(t) \\
\beta_2(t) \\
\beta_3(t)
\end{bmatrix}
= \begin{bmatrix}
0 \\
0 \\
0
\end{bmatrix}
\]  \hspace{1cm} (6.8)

where \( \rho \overset{\text{def}}{=} \frac{d}{dt} \) denotes the differential operator. As the degree \( A(\rho) = 4 \) the corresponding initial condition vector, as in (4.5), \( \beta(0-) \in \mathbb{R}^{12} \). Let us therefore consider a solution of the above problem under the following initial conditions

\[ \beta^{(1)}(0-) = 1, \quad \beta^{(j)}(0-) = 0 \quad \forall (i,j) \neq (2,1). \]  \hspace{1cm} (6.9)

This problem is implemented below.

```plaintext
> with(lineal):  
Warning: new definition for norm  
Warning: new definition for trace

> with(linsys):  

> readlib(laplace):  

> A:=matrix(3,3,[s,s^4,s^2+1,s^3,s^1,0,s^1,0]);
```
\[
A := \begin{bmatrix}
4 & 2 \\
s & s & s + s \\
1 & s & s + 1 \\
0 & s + 1 & 0
\end{bmatrix}
\]

> Bin:=vector(12,[0,0,0,0,1,0,0,0,0,0,0,0]);

\[
Bin := [ 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 ]
\]

> readlib(showtime): showtime();

01 :=

> ARSOLVE(A,Bin,4);

\[
\begin{bmatrix}
- \exp(-t) \sin(t) + \exp(-t) \cos(t) + ys[1] \Dirac(t) - ys[1] \exp(-t) \sin(t) \\
- ys[3] \exp(-t) \cos(t), \\
0, \Dirac(t) - \exp(-t) \sin(t) - \exp(-t) \cos(t) - ys[1] \exp(-t) \cos(t) \\
+ ys[3] \exp(-t) \sin(t)
\end{bmatrix}
\]

time 1.55 words 221260

02 :=

> off;

Consider the AR-Representation as in (6.8). However, now consider the solution under
the following initial conditions
\[
\beta^{(0)}_1(0-) = 1, \quad \beta^{(0)}_3(0-) = 1, \quad \beta^{(j)}_4(0-) = 0 \quad \forall (i,j) \neq (1,0) \text{ and } (3,0) \quad (6.10)
\]

> with(linalg):
Warning: new definition for norm
Warning: new definition for trace

> with(linsys):

> readlib(laplace):

> A:=matrix(3,3,[s,s-4,s-2+s,1,s-3,s+1,0,s+1,0]);

\[
A := \begin{bmatrix}
4 & 2 \\
s & s & s + s \\
1 & s & s + 1 \\
0 & s + 1 & 0
\end{bmatrix}
\]

> Bin:=vector(12,[1,0,1,0,0,0,0,0,0,0,0,0]);
Bin := \{ 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0 \}

> readlib(showtime): showtime();

01 :=

> ARSOLVE(A,Bin,4);

the system is not solvable

time  1.11  words  151340

02 :=

> off

This example was considered in Karampetakis (1995b) where it was shown that no solution existed under the initial conditions (6.10).

6.4. FEEDBACK PROBLEMS

Feedback Compensation

Consider an open-loop system with given transfer function \( G(s) \in \mathbb{R}(s)^{2\times 3} \)

\[
G(s) = \begin{pmatrix}
\frac{1}{s-1} & 0 & 1 \\
0 & \frac{1}{s-2} & 0
\end{pmatrix}.
\]  (6.11)

The Smith McMillan Form of \( G(s) \), denoted by \( S_M(G(s)) \) can be seen to be

\[
S_M(G(s)) = \begin{pmatrix}
1 & 0 & 0 \\
\frac{(s-1)(s-2)}{s^2} & 1 & 0 \\
0 & 1 & 0
\end{pmatrix}.
\]  (6.12)

Therefore \( G(s) \) has two finite poles at \( s = 1 \) and \( s = 2 \) and therefore the system is not stable. Let us apply output feedback \( F(s) \in \mathbb{R}(s)^{3\times 2} \) to the system, of the form (4.7), so as to stabilize it: i.e. to produce a closed loop transfer function matrix \( H(s) \in \mathbb{R}(s)^{2\times 3} \) with, say, a Smith McMillan Form

\[
S_M(H(s)) = \begin{pmatrix}
\frac{1}{2(s+1)} & 0 & 0 \\
0 & \frac{1}{2} & 0
\end{pmatrix}
\]  (6.13)

which has one finite pole at \( s = -1 \). Such a matrix \( H(s) \) can be seen to be

\[
H(s) = \begin{pmatrix}
\frac{1}{s+1} & \frac{1}{2(s+1)} & \frac{s-1}{s+1} \\
2 & \frac{s-1}{2(s+1)} & \frac{2(s-1)}{s+1}
\end{pmatrix}.
\]  (6.14)

Using the Maple procedure CONSOLVE given earlier we can compute such a general output feedback compensator. This is illustrated below.
> with(linalg):
Warning: new definition for norm
Warning: new definition for trace

> with(linsys):

> G := matrix(2, 3, [1/(s-1), 0, 1, 0, 1/(s-2), 0]);

\[
\begin{bmatrix}
1 & 0 & 1 \\
\frac{s}{s-1} & 1 & 0 \\
1 & 0 & 0 \\
\end{bmatrix}
\]

> H := matrix(2, 3, [-1/(s+1), 1/(2(s+1)), -\( \frac{s-1}{s+1} \), -2/(s+1), -1/2(s-1)/(s+1), -2*sqrt(s-1)/(s+1)]);

\[
\begin{bmatrix}
1 & 1 & s - 1 \\
\frac{s+1}{s+2} & 2 & s + 1 \\
2 & s - 1 & s - 1 \\
\frac{s+1}{s+1} & s + 1 & s + 1 \\
\end{bmatrix}
\]

> readlib(showtime): showtime();

01 :=

> CONSOLVE(G, H);

\[
\begin{align*}
2 & \frac{-2 s + 2 + Y[1, 1] s - 2 s Y[1, 1] + Y[1, 1] - Y[3, 1] s + Y[3, 1]}{2 + s - 2 s}, \\
\frac{-1 + s Y[1, 2] - 2 s Y[1, 2] + Y[1, 2] - Y[3, 2] s + Y[3, 2]}{2 + s - 2 s}, \\
\frac{Y[4, - 4 s + 2] - Y[3, 1] s + Y[1, 1] - Y[1, 1]}{2 + s - 2 s}, \\
\frac{2 s - 4 s + 2 - Y[3, 1] s + Y[1, 1] - Y[1, 1]}{2 + s - 2 s}, \\
\frac{s - 1 - Y[3, 2] s + Y[1, 2] - Y[1, 2]}{2 + s - 2 s}
\end{align*}
\]
It can be seen that under the parameter selection \( Y[1,1] = 0, Y[3,1] = -2, Y[1,2] = -1, Y[3,2] = 0 \) the feedback matrix obtained above reduces to

\[
F(s) = \begin{pmatrix} 0 & -1 \\ 4 & -s \\ -2 & 0 \end{pmatrix}
\]

(6.15)

which is polynomial. It is obvious that there exists no proper compensator as can be seen from element (2,2) above.

ii Feedback Matching

Consider an open-loop system with given transfer function \( G(s) \in \mathbb{R}(s)^{1 \times 2} \)

\[
G(s) = \begin{pmatrix} \frac{1}{s-1} \\ 2 \end{pmatrix}.
\]

(6.16)

The Smith McMillan Form of \( G(s) \), denoted by \( S_M(G(s)) \) can be seen to be

\[
S_M(G(s)) = \begin{pmatrix} \frac{1}{s-1} & 0 \\ \end{pmatrix}
\]

(6.17)

which implies that \( G(s) \) has one finite pole at \( s = 1 \) and is therefore not stable. Let us apply a pre-compensator \( K(s) \in \mathbb{R}(s)^{1 \times 2} \) and an output feedback \( F(s) \in \mathbb{R}(s)^{2 \times 1} \) to the system so as to stabilize it: i.e. to produce a closed-loop transfer function matrix \( H(s) \in \mathbb{R}(s)^{1 \times 2} \) with, say, a Smith McMillan Form

\[
S_M(H(s)) = \begin{pmatrix} \frac{1}{s+1} & 0 \end{pmatrix}.
\]

(6.18)

This has one finite pole at \( s = -1 \). Such a matrix \( H(s) \) can be seen to be of the form

\[
H(s) = \begin{pmatrix} 1 \\ \frac{1}{s+1} \end{pmatrix}.
\]

(6.19)

Using the Maple procedure MATCH given earlier we can compute such a compensator pair \( [F(s), K(s)] \) providing that such a pair exists. This is illustrated below.

> with(linalg):
Warning: new definition for norm
Warning: new definition for trace

> with(linsys):

> G:=matrix(1,2,[1/(s-1),2]);

\[
G := \begin{pmatrix} 1 \\ \frac{1}{s-1} \end{pmatrix}
\]

> H:=matrix(1,2,[1,1/(1+s)]);
\[ H := \begin{bmatrix} 1 \\ 1 \quad \cdots \\ 1 + s \end{bmatrix} \]

\[ F \text{ is given by,} \]
\[ \begin{array}{c}
\((- s - s + 2 + 13 Y(1, 1) - 6 Y(1, 1) s - 11 Y(1, 1) s^2 + 8 Y(1, 1) s^3 - 2 Y(2, 1) s^2 - 2 Y(2, 1) s^3 + 4 Y(2, 1) + Y(1, 2) + 2 Y(1, 2) s^2 + Y(1, 2) s^3 - 2 Y(2, 2) s^2 + 2 Y(2, 2) s^3 - 2 Y(2, 3) s^2 - 2 Y(2, 3)) / (5 + 4 s - 8 s) (3 + 4 s + 2 s)) \end{array} \]

\[ \begin{array}{c}
\((- 4 + 4 s + 4 Y(2, 3) s - 4 Y(2, 3) s + 2 s + 2 Y(1, 3) s + 4 Y(2, 2) s - 2 Y(1, 2) s - 2 Y(1, 1) s + 4 Y(2, 1) s - 2 s + 4 Y(1, 1) s - 2 Y(1, 3) + 7 Y(2, 1) + 4 Y(2, 2) + 4 Y(2, 3) - 4 Y(2, 3) s^2 + 4 Y(2, 1) s - 2 Y(1, 1) s - 6 Y(2, 1) s - 2 Y(1, 2) s + 2 Y(1, 2) s^2 - 8 Y(2, 2) s) / ((5 + 4 s - 8 s) (3 + 4 s + 2 s)) \end{array} \]

\[ K \text{ is given by,} \]
\[ \begin{array}{c}
\((- 1 + s - s + s + 8 Y(1, 2) s + Y(1, 1) + 13 Y(1, 2) + Y(1, 3) - 2 Y(2, 1) s + 4 Y(2, 2) - 2 Y(2, 3) + Y(1, 3) s + 2 Y(2, 3) s - 2 Y(2, 1) s + 2 Y(1, 1) s + Y(1, 1) s + 2 Y(2, 1) s + 2 Y(2, 1) s - 6 Y(1, 2) s) \end{array} \]
\[-11 Y[1, 2] s - 2 Y[2, 2] s - 2 Y[2, 1] s\]
\[
\begin{array}{c}
\text{//} \\
2 \\
\text{//}
\end{array}
\]
\[
\begin{array}{c}
\end{array}
\]
\[
\begin{array}{c}
\text{//} \\
2 \\
\text{//}
\end{array}
\]
\[
\begin{array}{c}
(5 + 4 s - 8 s) (3 + 4 s + 2 s) \\
3 + 4 s + 2 s
\end{array}
\]
\[
\begin{array}{c}
- 2 Y[1, 2] s + 2 Y[1, 1] s + 4 Y[2, 1] s + 2 s - 2 Y[1, 1] s \\
+ 4 Y[1, 2] s - 2 Y[1, 1] s + 2 Y[1, 1] s - 8 Y[2, 1] s - 2 Y[1, 2] s \\
- 6 Y[2, 2] s
\end{array}
\]
\[
\begin{array}{c}
(5 + 4 s - 8 s) (3 + 4 s + 2 s) \\
- (5 + 4 s - 8 s)
\end{array}
\]
\[
\begin{array}{c}
2 + 2 s + 4 Y[1, 3] s + 4 Y[2, 3] s + 2 s + 4 Y[1, 3] s + 2 Y[1, 1] s \\
\end{array}
\]
\[
\begin{array}{c}
(5 + 4 s - 8 s) (3 + 4 s + 2 s) \\
2 + 2 s + 4 Y[1, 2] s + Y[1, 1] + 13 Y[1, 2] + Y[1, 3] - 2 Y[2, 1] s \\
\end{array}
\]
\[
\begin{align*}
+ 2 \ Y[1, 1] s + Y[1, 1]^2 s & + 2 \ Y[2, 1] s + 2 \ Y[2, 1]^2 s - 6 \ Y[1, 2] s \\
- 11 \ Y[1, 2] s - 2 \ Y[2, 2] s - 2 \ Y[2, 2]^2 s & \\
/ & (5 + 4 s - 8 s)(3 + 4 s + 2 s), (s - 1 + 13 \ Y[1, 3]^2 s) \\
- 8 \ Y[1, 3] s - 12 \ Y[1, 3] s + 8 \ Y[1, 3]^3 s - 4 \ Y[2, 3] s & + 2 \ Y[2, 1] s \\
- 4 \ Y[2, 3] s + 2 \ Y[2, 2] s + 4 \ Y[1, 1] s + Y[1, 2] s + 4 \ Y[2, 3] s & \\
/ & (5 + 4 s - 8 s)(3 + 4 s + 2 s)) \\
[(2 + 4 \ Y[2, 3] s - 4 \ Y[2, 3] s - 4 s + 2 \ Y[1, 3] s + 4 \ Y[2, 2] s & \\
2 \ Y[1, 2] s + 2 \ Y[1, 1] s + 4 \ Y[2, 1] s + 2 s - 2 \ Y[1, 1] & \\
+ 4 \ Y[1, 2] s - 2 \ Y[1, 3] + 4 \ Y[2, 1] + 7 \ Y[2, 2] + 4 \ Y[2, 3] - 4 \ Y[2, 3] s & \\
+ 4 \ Y[2, 2] s - 2 \ Y[1, 1] s + 2 \ Y[1, 1] s - 8 \ Y[2, 1] s - 2 \ Y[1, 2] s & \\
- 6 Y[2, 2] s & / (5 + 4 s - 8 s)(3 + 4 s + 2 s)), (-2 - 2 s) \\
/ & + 2 s + 4 \ Y[1, 3] s + 4 \ Y[2, 3] s + 2 s + 4 \ Y[1, 3] s + 2 \ Y[1, 1] s \\
+ 2 \ Y[1, 2] - 4 \ Y[1, 3] - 4 \ Y[2, 1] - 4 \ Y[2, 2] - 7 \ Y[2, 3] - 4 \ Y[1, 3] s & \\
- 6 \ Y[2, 3] s + 4 \ Y[2, 1] s + 4 \ Y[2, 2] s - 2 \ Y[1, 1] s + 4 \ Y[2, 1] s & \\
- 4 \ Y[2, 1] s - 2 \ Y[1, 2] s + 4 \ Y[2, 2] s - 4 \ Y[2, 2] s & ) \\
/ & (5 + 4 s - 8 s)(3 + 4 s + 2 s)) \\
\text{time} & 1.65 \ \text{words} \ 229326 \ \\
02 := & \end{align*}
\]

It can be seen that under the parameter selection \(Y[1, 1] = 6, Y[1, 2] = 6, Y[1, 3] = 7\),
\[ Y[2,1] = 1, \quad Y[2,2] = 7/2, \quad Y[2,3] = 0 \]
the compensator pair \( F(s)K(s) \) reduce to

\[
F(s) = \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \quad K(s) = \begin{pmatrix} 6 & 3 \\ 3 & 0 \end{pmatrix}
\]

which are constant.

7. Conclusions

In this paper we have presented an algorithm for the computation of the generalized inverse of a singular rational matrix \( A(s) \in \mathbb{R}^{n \times m} \). This algorithm is a direct extension from Decell (1965) and Karampetakis (1995a) where the constant and polynomial matrix cases have been considered respectively; indeed the same algorithm can be seen to hold for when \( A(s) \) is either rational, polynomial or constant. The algorithm is recursive and can be implemented computationally; this has subsequently been done in the symbolic computational package Maple which enables polynomial operations in an indeterminate to be carried out. Subsequently, a corresponding Maple procedure for the implementation of the algorithm is given. Several applications of the generalized inverse in linear systems theory are shown and similarly, for each a corresponding Maple procedure is given. These procedures are implemented in Maple via the introduction of the linvsy package which is a package of procedures for use within linear systems. Finally, several examples have been considered to show the implementation of these routines in Maple and the results from these have been discussed. The clear benefit of computing such a generalized inverse is that it enables a wider set of such problems to be solved.

References


Originally received 19 February 1996
Accepted 19 February 1996