Computation of the Generalized Inverse of a Rational Matrix via Maple and Applications

J. Jones†, N. P. Karampetakis‡ & A. C. Pugh†

†Department of Mathematical Sciences
Loughborough University
Loughborough, Leics. LE11 3TU
England.
J.Jones1@Lboro.ac.uk

‡Department of Mathematics
Aristotle University of Thessaloniki
54006 Greece

Abstract

In [6] 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.

1. Introduction

Consider a matrix $A \in \mathbb{R}^{n\times m}$. For A non–singular (i.e. $n = m$ and $|A| \neq 0$) $\exists$ a unique non–singular matrix $B \in \mathbb{R}^{n\times n}$, termed the *inverse* of A, such that

$$AB = BA = I_n \tag{1}$$

where $I_n \stackrel{\text{def}}{=}$ the $(n \times n)$ identity matrix. 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

$$AC = I_n \quad (resp.\ DA = I_m) \tag{2}$$

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

Consider the solution of a set of consistent linear equations of the form

$$Ax = b \tag{3}$$

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

i) $rank[A] = n$ – $\exists$ a right inverse $C \in \mathbb{R}^{m\times n}$ s.t. (2) holds. The vector $x = Cb$ is a solution to (3) and is unique iff A is non–singular. (i.e. $C \equiv A^{-1}$).

ii) $rank[A] = m$ – $\exists$ a left inverse $D \in \mathbb{R}^{m\times n}$ s.t (2) holds. The vector $x = Db$ is a solution to (3) iff $ADb = b$ and is unique in this case.

iii) $rank[A] = k \leq min\{m, n\}$ – A solution of the form $x = Pb$ may still exist.

Subsequently a more "generalized" class of inverse has been defined [9] which enable such solutions to be obtained. Indeed the problem of computing such "generalized inverses" has been considered by many authors [1], [3], [5]–[9]. This is due to the large number of applications such an inverse has in linear systems theory which include the calculation of the transfer function of a matrix [4], the solution of linear equations and the solution of diophantine equations [5].

In [3] 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 [6] for the more general singular polynomial matrix case

$$A(s) \stackrel{\text{def}}{=} A_q s^q + \cdots + A_1 s + A_0 \in \mathbb{R}[s]^{n\times m} \tag{4}$$

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 $A(s) \in \mathbb{R}(s)^{n\times m}$.

In section 2 we define several classes of generalized inverse and following [6] present an algorithm in section 3 for the computation of a specific class of generalized inverse, denoted by $A(s)^\dagger \in \mathbb{R}(s)^{m\times n}$, for a matrix

$A(s) \in \mathbb{R}(s)^{n\times m}$. In section 4 this algorithm is implemented in the symbolic computational package Maple [2]. In sections 5 and 6 we consider an application of the generalized inverse in solving linear matrix equations and illustrate the implementation via a numerical example.

2. Preliminaries

Definition 1. 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:

$$\begin{aligned} &(i)\ AGA = A && (iii)\ \left[AG\right]^{\star} = AG \\ &(ii)\ GAG = G && (iv)\ \left[GA\right]^{\star} = GA \end{aligned} \tag{5}$$

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

Analogously we can define a generalized inverse of a polynomial matrix $A(s) \in \mathbb{R}[s]^{n\times m}$ and 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 correspondingly satisfies at least the first *or* the second conditions in (5).

Various classes of generalized inverses can be defined depending on which of conditions (5) they satisfy. It is usual to use the following notation [1].

Definition 2. Any matrix which satisfies equations (a), (b), ..., from among $(i)-(iv)$ in (5) will be called an $\{a, b, \ldots\}$–inverse of A and be denoted by $A^{(a,\ b,\ \ldots)}$.

Several classes of generalized inverse are as below.

$\{1\}$–**inverse** $\stackrel{\text{def}}{=} A^{(1)}$ – Used to solve a set of consistent linear equations as in (3).

$\{1,\ 3\}$–**inverse** $\stackrel{\text{def}}{=} A^{(1,\ 3)}$ – Used to form a least squares solution to a set of inconsistent linear equations (3).

$\{1,\ 2,\ 3\}$–**inverse** $\stackrel{\text{def}}{=} A^{(1,\ 2,\ 3)}$ – Every right inverse of A, as defined in (2), is a $\{1,\ 2,\ 3\}$–inverse of A.

$\{1,\ 2,\ 4\}$–**inverse** $\stackrel{\text{def}}{=} A^{(1,\ 2,\ 4)}$ – Every left inverse of A, as defined in (2), is a $\{1,\ 2,\ 4\}$–inverse of A.

$\{1,\ 2,\ 3,\ 4\}$–**inverse**, $A^{\dagger}$ – Used to form a minimum–norm least squares solution to a set of inconsistent linear equations (3).

The generalized inverse class, $A^{\dagger}$, is unique for each given matrix $A \in \mathbb{R}^{n\times m}$ and such a matrix always exists [9]. Further, it can be seen that

$$A^{\dagger} \subset A^{(1,\ 2,\ 3)} \subset \ldots \subset A^{(1)} \tag{6}$$

i.e. $A^{\dagger}$ belongs to *each* generalized inverse class. In th following paper we only consider the class $A^{\dagger}$ and defin this as "the generalized inverse".

3. Computation of the Generalized Inverse of $A(s) \in \mathbb{R}(s)^{n\times m}$

In this section we present an algorithm for the com putation of the generalized inverse, $A^{\dagger} \in \mathbb{R}(s)^{m\times n}$, c a singular rational matrix $A(s) \in \mathbb{R}(s)^{n\times m}$. The al gorithm can be seen to be highly computationally at tractive and the computational implications are subse quently discussed.

3.1. Algorithm

The algorithm is a direct extension of [6] for the singula polynomial matrix case. Here we consider s to be a*indeterminate* although the same algorithm holds fo when $s \in \mathbb{C}$ under some slight modifications.

Step 1: Consider $A(s) \in \mathbb{R}(s)^{n\times m}$ and form the fol lowing sequences

$$\left.\begin{aligned} &\left[A_0(s),\ A_1(s),\ \ldots,\ A_n(s)\right] \\ &\left[a_0(s),\ a_1(s),\ \ldots,\ a_n(s)\right] \\ &\left[B_0(s),\ B_1(s),\ \ldots,\ B_n(s)\right] \end{aligned}\right\} \tag{7}$$

which are constructed recursively as follows:

$$\begin{aligned} i = 0 \quad \Rightarrow \quad & \begin{cases} A_0(s) = 0_{n,n} \\ a_0(s) = 1 \\ B_0(s) = I_n \end{cases} \\ i = 1,\ \ldots,n \quad \Rightarrow \quad & \begin{cases} A_i(s) = \left[A(s)A(s)^T\right] B_{i-1}(s) \\ a_i(s) = -\dfrac{trace\,(A_i(s))}{i} \\ B_i(s) = A_i(s) + a_i(s) I_n \end{cases} \end{aligned} \tag{8}$$

Step 2: Let $a_n(s) = \ldots = a_{k+1}(s) = 0$ while $a_k(s) \neq 0$ Then the *Generalized Inverse*, $A(s)^{\dagger}$, of $A(s)$ is given by

$$A(s)^{\dagger} \stackrel{\text{def}}{=} -a_k(s)^{-1} A(s)^T B_{k-1}(s) \tag{9a}$$

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

$$A(s)^{\dagger} \stackrel{\text{def}}{=} 0_{m,n} \tag{9b}$$

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 3 sequences are added to at each step. Hence it is very *computationally attractive*.

2. The storage requirement can be significantly reduced via two means. Firstly 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 $[A(s)A(s)^T] \in \mathbb{R}[s]^{n\times n}$ in (8) is constant $\forall i \geq 1$ and subsequently only needs to be calculated once (i.e. outside the loop).

6. For $n > m$ we can form the transpose of $A(s)$, denoted by $A^T(s) \in \mathbb{R}(s)^{m\times n}$, and compute the generalized inverse of this matrix instead. Then $A(s)^\dagger = \left[A^T(s)^\dagger\right]^T$. This will in general be quicker to compute as the algorithm will now be one of m rather than n steps.

7. The same *algorithm* holds for when $A(s)$ is rational, polynomial or indeed constant. It also holds for n-D matrices of the form $A(s_i)$, $i = 1, \ldots, n$ [7]. Therefore only one computer procedure is required.

4. Implementation via Maple

In this section we implement Algorithm 3.1. in the symbolic computational package Maple [2] via the introduction of a procedure `GINVERSE`. The issue concerning the computational time for computing generalized inverses is subsequently addressed for this procedure.

4.1. 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 s.

Procedures – Inbuilt procedures exist which can be implemented directly. In Maple the `linalg` package consists of numerous 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 can be viewed upon as *building blocks* for more specialized and advanced procedures.

4.2. The `linsol` 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

$$\texttt{> with}(package);$$

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

$$> file1(args1);$$

The `linsol` package is such a package which is concerned with the solution of linear systems. The package contains *two* procedures, both based on Algorithm 3.1., for the computation of the generalized inverse, $A(s)^\dagger \in \mathbb{R}(s)^{m\times n}$, of a given matrix $A(s) \in \mathbb{R}(s)^{n\times m}$.

1. `GINVERSE` – Forms $A(s)^\dagger \in \mathbb{R}(s)^{m\times n}$ where s is considered to be an *indeterminate*.

2. `GINVcomp` – Forms $A(s)^\dagger \in \mathbb{R}(s)^{m\times n}$ where $s \in \mathbb{C}$.

The difference between these procedures can be illustrated by considering the matrix

$$A(s) = \begin{bmatrix} 1 \\ s \end{bmatrix} \tag{10}$$

Implementing the two procedures results in

$$\begin{aligned} \texttt{GINVERSE},\ A(s)^\dagger &= \begin{bmatrix} \frac{1}{1+s^2} & \frac{s}{1+s^2} \end{bmatrix} \\ \texttt{GINVcomp},\ A(s)^\dagger &= \begin{bmatrix} \frac{1}{1+s\bar{s}} & \frac{\bar{s}}{1+s\bar{s}} \end{bmatrix} \end{aligned} \tag{11}$$

where $\bar{s}$ denotes the conjugate of s. In this paper we will only consider the procedure `GINVERSE`.

The time for computing the generalized inverse via Algorithm 3.1. is clearly dependent on the dimension of the matrix concerned; corresponding to a matrix $A(s) \in \mathbb{R}(s)^{n\times m}$ the algorithm will comprise of $min\{n,\ m\}$ steps (from 3.2.(6)). Let us consider the non-square

polynomial matrix $A(s)$ of degree deg, where

$$A(s) = \begin{bmatrix} 1 & s^{deg} & & \\ & 1 & \ddots & \\ & & \ddots & s^{deg} \\ & & & 1 \quad s^{deg} \end{bmatrix} \in \mathbb{R}[s]^{n\times(n+1)} \tag{12}$$

where $n = 1, \ldots, 24$. The respective times to compute the generalized inverse of a set of matrices (12) of fixed degree deg is illustrated in the figures below. In Fig.1. we have considered $deg = 5$ while in Fig.2. we have considered the cases $deg = 5,\ 10,\ 15,\ 20,\ 25,\ 30$. Both show that the computational time depends explicitly on the dimension n of the matrix.

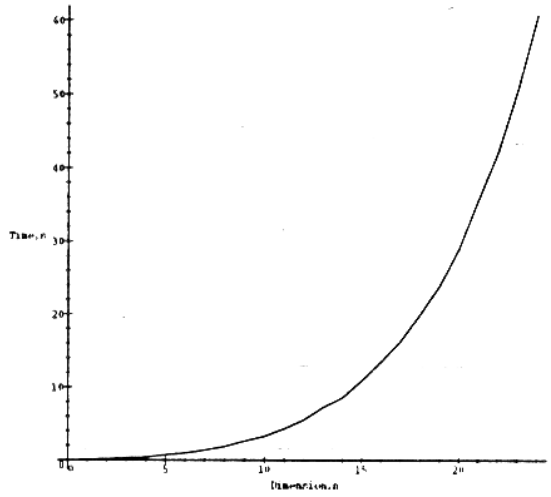

Fig. 1. Computational Time ($deg = 5$)

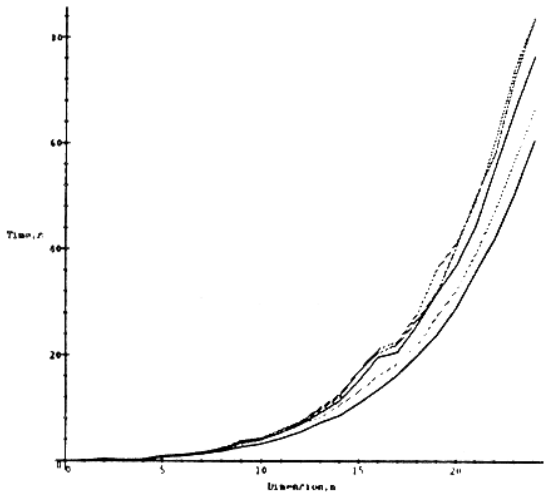

Fig. 2. Computational Time ($deg = 5,\ 10,\ \ldots,\ 30$) (Increasing Degree)

5. Applications

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

$$A(s)X(s)B(s) = C(s) \tag{13}$$

where $A(s) \in \mathbb{R}(s)^{n\times m}$, $B(s) \in \mathbb{R}(s)^{k\times l}$ and $C(s) \in \mathbb{R}(s)^{n\times l}$. From [8] we have the following

Theorem 1. 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 $C(s) \in \mathbb{R}(s)^{n\times l}$ has a solution $X(s) \in \mathbb{R}(s)^{m\times k}$ iff the consistency condition

$$A(s)A(s)^{\dagger}C(s)B(s)^{\dagger}B(s) = C(s) \tag{14a}$$

is satisfied in which case all the solutions are given by

$$\begin{aligned} X(s) = \quad & A(s)^{\dagger}C(s)B(s)^{\dagger} + Y(s) \\ & -A(s)^{\dagger}A(s)Y(s)B(s)B(s)^{\dagger} \end{aligned} \tag{14b}$$

where $A(s)^{\dagger}$ and $B(s)^{\dagger}$ are the generalized inverses o $A(s)$ and $B(s)$ respectively and $Y(s)$ is arbitrary t within having the same dimension as $X(s)$. □

The above Theorem similarly holds for whe $A(s)^{\dagger}$, $B(s)^{\dagger}$ are replaced with specific $\{1\}$–inverses $A^{(1)}$, $B^{(1)}$ respectively.

We can apply the above Theorem to several areas Clearly for the case $B(s) = C(s) = A(s)$, (14b) gives al possible $\{1\}$–inverses of $A(s)$ given a single $\{1\}$–inverse we can form other classes of generalized inverses in a analogous manner. Other applications include feedback compensation and matching problems [5], the solutio of AR–Representations [6] and the solution of diophantine equations which we will consider further in the next section.

A procedure `PLINSOLVE` has been written in Maple fo the solution of the matrix equation (13) based on Theorem 1. This is included in the `linsol` package as defined in section 4. Similarly procedures have been written for feedback compensation, feedback matching an AR–Representation problems [5].

6. Example

Consider the polynomial matrix diophantine equation

$$\begin{bmatrix} 1 & 0 \\ 0 & s \end{bmatrix} W(s) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} Y(s) = \begin{bmatrix} s & 1 \\ s & 0 \end{bmatrix} \tag{15}$$

and write it in the form

$$\underbrace{\left[\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & s & 0 \end{array}\right]}_{A(s)} \underbrace{\begin{bmatrix} W(s) \\ \hline Y(s) \end{bmatrix}}_{X(s)} = \underbrace{\begin{bmatrix} s & 1 \\ s & 0 \end{bmatrix}}_{C(s)} \tag{16}$$

Clearly (16) is of the form (13) where $A(s) \in \mathbb{R}[s]^{2\times 3}$, $B(s) \equiv I_2$ and $C(s) \in \mathbb{R}[s]^{2\times 2}$ and we require to solve (16) for a polynomial solution pair $W(s) \in \mathbb{R}[s]^{2\times 2}$, $Y(s) \in \mathbb{R}[s]^{1\times 2}$. It can be seen that the reduced consistency condition (14a) is satisfied and therefore a solution to (16) exists. This solution will be polynomial as every left divisor of $A(s)$ in (16) is also a left divisor of $C(s)$.

We have 2 methods to solve (16) 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-parameterized* for $k > 1$. Given $A(s)X(s) = C(s)$, where $A(s) \in \mathbb{R}[s]^{n\times m}$ $(rank[A(s)] = r)$, $X(s) \in \mathbb{R}[s]^{m\times k}$ 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–parameterized. The **`PLINSOLVE`** procedure, when applied to the equation $A(s)X(s) = C(s)$, is generally *over–parameterized* 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.

6.1. Maple Worksheet

Here we implement the Maple procedures **`PLINSOLVE`** and **`linsolve`** to solve the linear matrix equation (16). 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.

```
libname:=/usr/local/maple/lib, /home/Ice
/majj2/mymaplelib

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

> with(linsol):

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

$$AB := \begin{bmatrix} 1 & 0 & 1 \\ 0 & s & 0 \end{bmatrix}$$

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

$$C := \begin{bmatrix} s & 1 \\ s & 0 \end{bmatrix}$$

```
> readlib(showtime):  showtime();

O1 :=

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

$$sol1 := \begin{bmatrix} \frac{1}{2}s + \frac{1}{2}Y_{1,1} - \frac{1}{2}Y_{3,1} & \frac{1}{2} + \frac{1}{2}Y_{1,2} - \frac{1}{2}Y_{3,2} \\ 1 & 0 \\ \frac{1}{2}s + \frac{1}{2}Y_{3,1} - \frac{1}{2}Y_{1,1} & \frac{1}{2} + \frac{1}{2}Y_{3,2} - \frac{1}{2}Y_{1,2} \end{bmatrix}$$

```
time 0.45 words 120258
O2 :=

> sol2:=linsolve(AB,C);
```

$$sol2 := \begin{bmatrix} s - \_t_1 & 1 - \_t_1 \\ 1 & 0 \\ \_t_1 & \_t_1 \end{bmatrix}$$

```
time 0.11 words 21591
O3 :=

> off
```

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 4 parameters $Y_{1,1}$, $Y_{3,1}$, $Y_{1,2}$, $Y_{3,2}$ (i.e. it is over–parameterized) while the **`linsolve`** output, $sol2$, contains only 1 parameter $\_t_1$ (i.e. it is under–parameterized).

For the substitution

$$p = Y_{1,1} - Y_{3,1}\ ,\ q = Y_{1,2} - Y_{3,2} \qquad (17)$$

$sol1$ takes the form

$$sol1 = \begin{bmatrix} \frac{1}{2}(s+p) & \frac{1}{2}(1+q) \\ 1 & 0 \\ \frac{1}{2}(s-p) & \frac{1}{2}(1-q) \end{bmatrix} \tag{18}$$

which contains the 2 independent parameters p and q as required.

In $sol2$ the parameter λ_1 is repeated in both of its columns. We can therefore form a complete solution by replacing λ_1 in the second column by a second independent parameter, λ_2, resulting in

$$sol2 = \begin{bmatrix} s-\lambda_1 & 1-\lambda_2 \\ 1 & 0 \\ \lambda_1 & \lambda_2 \end{bmatrix} \tag{19}$$

The two solutions (18) and (19) can be seen to be identical under the parameter relation

$$\lambda_1 = \frac{1}{2}(s-p) \ , \ \lambda_2 = \frac{1}{2}(1-q) \tag{20}$$

7. Conclusions

In this paper we have presented an algorithm for the computation of the generalized inverse of a singular rational matrix $A(s) \in \mathbb{R}(s)^{n\times m}$. This algorithm is a direct extension from [3] and [6] where the constant and polynomial matrix cases have been considered respectively. 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. The implementation of the corresponding procedures in Maple has been illustrated via a numerical example. In the case of solving consistent linear matrix equations the respective generalized inverse based procedure can be seen to give an equivalent solution as that of the inherent Maple procedure. The clear benefit of computing such a generalized inverse is that it enables a wider set of such problems to be solved.

NOTE: Due to the page restriction, the documented Maple code has not been included here. This can be obtained from contacting the authors directly.

References

[1] A. Ben-Israel, and T. N. E. Greville, *Generalized Inverses, Theory and Applications*. New York: Wiley, 1974.

[2] B. W. Char, K. C. Geddes, G. H. Gonnet, and S. M. Watt, *Maple V Language Reference Manual*. Springer-Verlag, 1991.

[3] H. P. Decell, "An application of the Cayley-Hamilton theorem to generalized matrix inversion," *SIAM review*, vol. 7, no. 4, pp. 526-528, 1965.

[4] G. Fragulis, B. G. Mertzios, and A. I. G. Vardulakis, "Computation of the inverse of a polynomial matrix and evaluation of its laurent expansion," *International Journal of Control*, vol. 53, pp. 431-443, 1991.

[5] J. Jones, N. P. Karampetakis, and A. C. Pugh, N.P. and Pugh, "The computation and applications of the generalized inverse via Maple," *Submitted for publication in the Journal of Symbolic Computation*, 1996.

[6] N. P. Karampetakis, "Computation of the generalized inverse of a polynomial matrix and applications," *To appear in Linear Algebra and its Applications*, 1995.

[7] N. P. Karampetakis, "Generalized inverses of two variable polynomial matrices and applications,". *Mathematical Sciences Report No. A247, Loughborough University of Technology*, 1995.

[8] V. Lovass-Nagy, R. J. Miller, and D. L. Powers, "An introduction to the simplest matrix-generalized inverse in systems science," *IEEE Transactions on Circuits and Systems*, vol. cas-25, no. 9, pp. 766-771, 1978.

[9] R. Penrose, "A generalized inverse for matrices," *Proceedings of the Cambridge Philosophical Society*, vol. 51, pp. 406-413, 1955.