Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

ELA

FAST COMPUTING OF THE MOORE-PENROSE INVERSE MATRIX*

VASILIOS N. KATSIKIS† AND DIMITRIOS PAPPAS‡

Abstract. In this article a fast computational method is provided in order to calculate the Moore-Penrose inverse of full rank $m \times n$ matrices and of square matrices with at least one zero row or column. Sufficient conditions are also given for special type products of square matrices so that the reverse order law for the Moore-Penrose inverse is satisfied.

Key words. Moore-Penrose inverse matrix, Computational methods, Tensor-product matrix, Reverse order law.

AMS subject classifications. 15A09.

1. Introduction. Let T be a $n \times n$ real matrix. It is known that when T is singular, then its unique generalized inverse $T^{\dagger}$ (known as the Moore-Penrose inverse) is defined. In the case when T is a real $m \times n$ matrix, Penrose showed that there is a unique matrix satisfying the four Penrose equations, called the generalized inverse of T. A lot of work concerning generalized inverses has been carried out, in finite and infinite dimension (e.g., [2, 11]).

In this article, we provide a method for the fast computation of the generalized inverse of full rank matrices and of square matrices with at least one zero row or column. In order to reach our goal, we use a special type of tensor product of two vectors, that is usually used in infinite dimensional Hilbert spaces. Using this type of tensor product, we also give sufficient conditions for products of square matrices so that the reverse order law for the Moore-Penrose inverse ([1, 4, 5]) is satisfied.

There are several methods for computing the Moore-Penrose inverse matrix (cf. [2]). One of the most commonly used methods is the Singular Value Decomposition (SVD) method. This method is very accurate but also time-intensive since it requires a large amount of computational resources, especially in the case of large matrices. In the recent work of P. Courrieu [3], an algorithm for fast computation of Moore-Penrose inverse matrices is presented based on a known reverse order law (eq. 3.2

*Received by the editors May 26, 2008. Accepted for publication November 20, 2008. Handling Editor: Michael Tsatsomeros.

†Department of Mathematics, National Technical University of Athens, Zographou 15780, Athens, Greece (vaskats@mail.ntua.gr, vaskats@gmail.com). Supported during his postdoctoral studies by the State Scholarship Foundation (IKY).

‡Athens University of Economics and Business, Department of Statistics, 76 Patission St., GR10434, Athens, Greece (dpappas@aueb.gr).

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

from [11]), and on a full-rank Cholesky factorization of possibly singular symmetric positive matrices. This is a fast algorithmic process; however, even in the case of rank deficient matrices, which is the proposed case by the author for the application of this method, the computation error is large compared to that of the SVD method[1] (see Table 4.2).

In the present manuscript, we construct a very fast and reliable method (see the **ginv** function in the Appendix) in order to estimate the Moore-Penrose inverse matrix of a rank-n tensor-product matrix. The computational effort required for the **ginv** function (see Figure 4.1) in order to obtain the generalized inverse is substantially lower, particularly for large matrices, compared to those provided by the other two methods (the SVD method and Courrieu's method). In addition, we obtain reliable and very accurate approximations in each one of the tested cases (Table 4.2). Also, from Theorem 3.1, that will be shown in paragraph 3, it is evident that the proposed method (**ginv** function), can also be used in the case of full-rank rectangular matrices. In what follows, we make use of the high-level language Matlab both for calculations of the generalized inverse of a tensor-product matrix, as well as for testing the reliability of the obtained results. Specifically, the Matlab 7.4 (R2007a) [9, 10] Service Pack 3 version of the software was used on an AMD Athlon(tm) 64 Processor 3000+ system running at 1.81GHz with 1.5 GB of RAM memory using the Windows XP Professional Version 2002 Service Pack 2 Operating System.

2. Preliminaries and notation. We shall denote by $\mathbb{R}^{m\times n}$ the linear space of all $m \times n$ real matrices. For $T \in \mathbb{R}^{m\times n}$, $R(T)$ will denote the range of T and $N(T)$ the kernel of T. The generalized inverse $T^\dagger$ (known as the Moore- Penrose inverse) is the unique matrix that satisfies the following four Penrose equations:

$$TT^\dagger = (TT^\dagger)^*, \qquad T^\dagger T = (T^\dagger T)^*, \qquad TT^\dagger T = T, \qquad T^\dagger TT^\dagger = T^\dagger,$$

where T^* denotes the transpose matrix of T.

It is easy to see that $\mathcal{R}(T^\dagger) = \mathcal{N}(T)^\perp$, where $TT^\dagger$ is the orthogonal projection onto $\mathcal{R}(T)$, and that $T^\dagger T$ is the orthogonal projection onto $\mathcal{N}(T)^\perp$. It is well known that $\mathcal{R}(T^\dagger) = \mathcal{R}(T^*)$. The number $r = \dim R(T)$ is called the rank of T and shall be denoted by $r(T)$ and $\langle\, , \rangle$ denotes the usual inner-product in $\mathbb{R}^n$.

3. The generalized inverse of a tensor-product matrix. According to [12], for each $x \in \mathbb{R}^k$, we consider the mapping

$$e \otimes f : \mathbb{R}^k \to \mathbb{R}^k \ \textit{with}\ (e \otimes f)(x) = \langle x, e\rangle f,$$

and assume that $\{e_1, ..., e_n\}$ and $\{f_1, ..., f_n\}$ are two collections of orthonormal vectors and linearly independent vectors of $\mathbb{R}^k, n < k$, respectively. Then, every rank-n

[1] The SVD method is implemented in the **pinv** function of Matlab.

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

operator T can be written in the following form

$$T = \sum_{i=1}^{n} e_i \otimes f_i.$$

In the following, we shall refer to this type of tensor product as the *tensor-product of the collections* $\{e_1, ..., e_n\}$ and $\{f_1, ..., f_n\}$. The adjoint operator T^* of T is the rank-n operator $T^* = \sum_{i=1}^{n} f_i \otimes e_i$.

The tensor-product of two collections of vectors, as defined above, is a linear operator. Therefore, it has a corresponding matrix representation T. In order to describe this representation, let us denote by $\{e_1, ..., e_n\}$ the first n vectors of the standard basis of $\mathbb{R}^k$, and suppose that the f_i's are in the form :

$$f_i = (f_{i1}, f_{i2}, ..., f_{ik}), i = 1, 2, ..., n$$

then the corresponding matrix T has the vectors f_i as columns. So,

$$T = \begin{pmatrix} f_{11} & f_{21} & ... & f_{n1} & 0 & ... & 0 \\ f_{12} & f_{22} & ... & f_{n2} & 0 & ... & 0 \\ \vdots & \vdots & ... & \vdots & \vdots & \vdots & \vdots \\ f_{1k} & f_{2k} & ... & f_{nk} & 0 & ... & 0 \end{pmatrix}.$$

We shall refer to this matrix T as the *tensor-product matrix* of the given collections.

In order to compute the Moore-Penrose inverse of the corresponding tensor-product matrix, we use the following theorem. For the sake of completeness we give a sketch of its proof.

THEOREM 3.1. [8, *Theorem 3.2*] *Let $\mathcal{H}$ be a Hilbert space. If $T = \sum_{i=1}^{n} e_i \otimes f_i$ is a rank-n operator then its generalized inverse is also a rank-n operator and for each $x \in \mathcal{H}$, it is defined by the relation*

$$T^\dagger x = \sum_{i=1}^{n} \lambda_i(x) e_i,$$

where the functions λ_i are the solution of an appropriately defined $n \times n$ linear system.

Proof. If $T = \sum_{i=1}^{n} e_i \otimes f_i$ then $\mathcal{R}(T) = [f_1, f_2, ..., f_n]$[2] and $\mathcal{R}(T^\dagger) = \mathcal{R}(T^*) = [e_1, e_2, ..., e_n]$. Therefore, for each $x \in \mathcal{H}$, we have $T^\dagger x = \sum_{i=1}^{n} \lambda_i(x) e_i$. Hence, in order to determine $T^\dagger$ one must calculate the functions λ_i, $i = 1, 2, ..., n$. It holds that $T^* x = T^* T T^\dagger x$, thus we have

[2] $[f_1, ..., f_n]$ denotes the closed linear span generated by the vectors $f_1, ..., f_n$.

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

$$\sum_{i=1}^{n}\langle x, f_i\rangle e_i = T^*x = T^*TT^\dagger x = \sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_j(x)\langle f_i, f_j\rangle e_i.$$

The last relation leads to the following $n \times n$ linear system:

$$\langle x, f_i\rangle = \sum_{j=1}^{n}\lambda_j(x)\langle f_i, f_j\rangle, \;\; i = 1, 2, ..., n.$$

The determinant of the above system is the Gram determinant of the linearly independent vectors $f_1, ..., f_n$ and hence, for each $x \in \mathcal{H}$, it has a unique solution where the unknowns are the functions λ_i, $i = 1, 2, ..., n$. □

Hence, one has to derive a procedure that will accurately implement the ideas of Theorem 3.1 in order to determine the generalized inverse of a tensor-product matrix. In other words, our main concern is to calculate the corresponding λ_i in the expansion

$$T^\dagger x = \sum_{i=1}^{n}\lambda_i(x)e_i$$

so that we can provide the generalized inverse $T^\dagger$. In order to reach our goal, the high-level computational environment of Matlab is employed in our study.

4. The computational method .

4.1. Method presentation and examples. The first step of our approach consists of constructing two functions named **ginv** and **ginvtest** (see Appendix). The function **ginv** first calculates the corresponding Gram matrix of the linearly independent vectors $f_1, ..., f_n$[3] and then it solves the appropriately defined $n \times n$ linear system. In particular, for each $j = 1, ..., n$, the **ginv** function provides the corresponding $\lambda_i(e_j)$ (see Theorem 3.1) in the expansion

$$T^\dagger e_j = \sum_{i=1}^{n}\lambda_i(e_j)e_i \tag{4.1}$$

in order to determine the generalized inverse of a given tensor-product matrix T. Therefore, for each $j = 1, 2, ..., n$ we have $\lambda_1(e_j), \lambda_2(e_j), ..., \lambda_n(e_j)$ as a solution of the corresponding linear system. Then, from (4.1) the generalized inverse $T^\dagger$ is having the following form

[3]That is, a $n \times n$ matrix $[a_{ij}]$, where $a_{ij} = \langle f_i, f_j\rangle$, for each $i, j = 1, ..., n$.

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

$$T^{\dagger} = \begin{pmatrix} \lambda_1(e_1) & \lambda_1(e_2) & ... & \lambda_1(e_k) \\ \lambda_2(e_1) & \lambda_2(e_2) & ... & \lambda_2(e_k) \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_n(e_1) & \lambda_n(e_2) & ... & \lambda_n(e_k) \\ 0 & 0 & ... & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & ... & 0 \end{pmatrix}.$$

Suppose that T is the corresponding matrix representation of a rank-n operator, then T is a $k \times k$ matrix whose first n columns are linearly independent vectors of $\mathbb{R}^k, n < k$ and all the other columns consists of zeros.

REMARK 4.1. It is clear, from Theorem 3.1, that our computational method will restrict in the case of full-rank rectangular and rank-deficient square matrices.

The **ginvtest** function allows us to compare the accuracy of the proposed method (**ginv** function) to that of the SVD method (Matlab **pinv** function) and on a recent, fast computational method by P. Courrieu [3] (**geninv** function). The **geninv** function is based on a known reverse order law (eq. 3.2 from [11]) and on a full-rank Cholesky factorization of possibly singular symmetric positive matrices. The accuracy of the results was examined in error matrices, with the matrix 2-norm, corresponding to the four properties characterizing the Moore-Penrose inverse (i.e., $TT^{\dagger} = (TT^{\dagger})^*, T^{\dagger}T = (T^{\dagger}T)^*, TT^{\dagger}T = T, T^{\dagger}TT^{\dagger} = T^{\dagger}$). The results of the **ginvtest** function are organized in a 3×4 matrix. The first row of this matrix contains the error results for the SVD method, the second row contains the error results for the Courrieu's method and the third row contains the error results of the proposed method.

The computational effort required to obtain the generalized inverse of a tensor-product matrix under different parameter configurations (i.e., number of vectors and dimensions) of the **ginv** function is substantially less. This is particularly true for large matrices, when a comparison is made between the results provided by the proposed method and those provided by the other two methods (SVD method, Courrieu's method). In particular, we have used the Matlab function **rand** in order to produce $m \times n$ matrices of values derived by a pseudorandom, scalar value drawn from a uniform distribution in the unit interval. Functions **ginv** and **ginvtest** (see Appendix) must be stored for further use. In our work, we stored them in a Matlab-accessible directory named **work**. Note that, **ginv** and **ginvtest** define functions that accept an input A, where A denotes a full rank matrix, in a different case the program responds with an explicit warning. We illustrate an example for a rank-8 tensor-product matrix, where $e_i, f_i \in \mathbb{R}^9, i = 1, 2, ..., 9$. For the purpose of monitoring the performance, we also present the execution times as well as the accuracy of the proposed method, the

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

SVD method and Courrieu's method (Table 4.1). The execution times have been recorded using the Matlab function **profile**.

EXAMPLE 4.2. Consider the tensor-product matrix $T = \sum_{i=1}^{8} e_i \otimes f_i$, generated by the linearly independent vectors,

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \end{pmatrix} = \begin{pmatrix} \frac{11983}{895} & \frac{5634}{239} & \frac{4018}{211} & \frac{3791}{336} & \frac{3661}{270} & \frac{2806}{139} & \frac{2675}{233} & \frac{3587}{206} & \frac{3995}{382} \\ \frac{12950}{1031} & \frac{1951}{79} & \frac{4361}{211} & \frac{3331}{239} & \frac{2735}{162} & \frac{3553}{223} & \frac{1936}{139} & \frac{10214}{471} & \frac{3809}{180} \\ \frac{1972}{147} & \frac{4892}{295} & \frac{2492}{187} & \frac{4337}{197} & \frac{4327}{177} & \frac{4700}{303} & \frac{9409}{626} & \frac{6570}{317} & \frac{25988}{1485} \\ \frac{2563}{155} & \frac{31967}{2740} & \frac{2611}{222} & \frac{2453}{235} & \frac{5224}{287} & \frac{7297}{294} & \frac{4948}{245} & \frac{10341}{439} & \frac{2941}{171} \\ \frac{36945}{2519} & \frac{430}{31} & \frac{5361}{371} & \frac{3207}{134} & \frac{5648}{317} & \frac{4237}{401} & \frac{4241}{352} & \frac{10467}{448} & \frac{5657}{240} \\ \frac{6869}{288} & \frac{3823}{237} & \frac{7583}{513} & \frac{2326}{111} & \frac{3099}{230} & \frac{4865}{209} & \frac{8140}{391} & \frac{3618}{241} & \frac{4787}{250} \\ \frac{1053}{64} & \frac{3955}{209} & \frac{1309}{80} & \frac{11004}{635} & \frac{43507}{2510} & \frac{3389}{143} & \frac{3260}{281} & \frac{2724}{133} & \frac{3853}{200} \\ \frac{3589}{281} & \frac{5838}{419} & \frac{2167}{123} & \frac{3885}{208} & \frac{2575}{133} & \frac{6517}{297} & \frac{6853}{346} & \frac{3553}{274} & \frac{4647}{203} \end{pmatrix}$$

and the vectors $e_1, ..., e_8$ of the standard basis of $\mathbb{R}^9$.

Then, T can be represented as a 9×9 matrix whose first 8 columns are the vectors $f_1, f_2, ..., f_8$ and the last one (column) consists of zeros.

We proceed with computing the Moore-Penrose inverse of T as follows :

In the command window of Matlab we type a matrix A that contains the nonzero block of T, i.e. A has the vectors $f_i, i = 1, 2, ..., 8$ as columns. Then, we invoke the **ginv** function by typing in the command window:

```
>> ginv(A)
```

The results, then, are as follows:

```
ans =
-912/17887  -129/1654    332/1989    340/4663   -169/2955    217/13563   232/7799    104/1645   -375/2668
 229/6452    311/2685   -563/5505   -145/1541    122/4031   -242/4405    135/11093  -504/11765   443/4180
 270/14543   226/3443    -93/988     -52/21341   206/2135   -209/8221    114/18613  -203/5305   -190/11351
-65/10176   -287/7155    140/22409  -131/4034    -35/5366     55/7514    683/18664   180/3301   -163/11332
-133/4162   -82/817      455/3792    143/2056   -182/2449   -67/4569     55/11147    218/2649   -128/2899
 284/3935    346/7209   -177/2984   -130/30261  -166/12675   -73/2878    243/9074   -107/2715    235/16527
 250/10141   115/2947   -489/5335   -175/9103    64/2221     264/3419   -96/745     -412/11707   282/3449
-695/10367  -173/2828    263/2961    94/3353    -108/14549   95/3309     224/9369   -130/5609   -32/18721
```

therefore the Moore-Penrose inverse $T^{\dagger}$ of T is the following matrix:

$$T^{\dagger} = \begin{pmatrix} \frac{-912}{17887} & \frac{-129}{1654} & \frac{332}{1989} & \frac{340}{4663} & \frac{-169}{2955} & \frac{217}{13563} & \frac{232}{7799} & \frac{104}{1645} & \frac{-375}{2668} \\ \frac{229}{6452} & \frac{311}{2685} & \frac{-563}{5505} & \frac{-145}{1541} & \frac{122}{4031} & \frac{-242}{4405} & \frac{135}{11093} & \frac{-504}{11765} & \frac{443}{4180} \\ \frac{270}{14543} & \frac{226}{3443} & \frac{-93}{988} & \frac{-52}{21341} & \frac{206}{2135} & \frac{-209}{8221} & \frac{114}{18613} & \frac{-203}{5305} & \frac{-190}{11351} \\ \frac{-65}{10176} & \frac{-287}{7155} & \frac{140}{22409} & \frac{-131}{4034} & \frac{-35}{5366} & \frac{55}{7514} & \frac{683}{18664} & \frac{180}{3301} & \frac{-163}{11332} \\ \frac{-133}{4162} & \frac{-82}{817} & \frac{455}{3792} & \frac{143}{2056} & \frac{-182}{2449} & \frac{-67}{4569} & \frac{55}{11147} & \frac{218}{2649} & \frac{-128}{2899} \\ \frac{284}{3935} & \frac{346}{7209} & \frac{-177}{2984} & \frac{-130}{30261} & \frac{-166}{12675} & \frac{-73}{2878} & \frac{243}{9074} & \frac{-107}{2715} & \frac{235}{16527} \\ \frac{250}{10141} & \frac{115}{2947} & \frac{-489}{5335} & \frac{-175}{9103} & \frac{64}{2221} & \frac{264}{3419} & \frac{-96}{745} & \frac{-412}{11707} & \frac{282}{3449} \\ \frac{-695}{10367} & \frac{-173}{2828} & \frac{263}{2961} & \frac{94}{3353} & \frac{-108}{14549} & \frac{95}{3309} & \frac{224}{9369} & \frac{-130}{5609} & \frac{-32}{18721} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

In order to test the accuracy of the three methods (i.e., SVD method (pinv), Courrieu's method (geninv) and the proposed method (ginv)) we invoke the **ginvtest** function by using the command:

```
>> ginvtest(A)
```

The cumulative results (execution times, accuracy) are presented in Table 4.1.

TABLE 4.1
Results for Example 4.2

	Time (seconds)	$\|TT^\dagger T - T\|_2$	$\|T^\dagger TT^\dagger - T^\dagger\|_2$	$\|TT^\dagger - (TT^\dagger)^*\|_2$	$\|T^\dagger T - (T^\dagger T)^*\|_2$
SVD method (Matlab pinv)	0.004	6.7251×10^{-13}	9.3345×10^{-16}	1.4001×10^{-14}	5.6939×10^{-14}
Courrieu's method (geninv)	0.005	4.0811×10^{-12}	2.1126×10^{-14}	2.2553×10^{-12}	8.5285×10^{-14}
Proposed method (ginv)	0.002	9.0576×10^{-15}	1.0375×10^{-14}	1.5797×10^{-15}	7.872×10^{-14}

It is apparent that the **ginv** function provides a practical, accurate and substantially faster numerical way for the calculation of the Moore-Penrose inverse of a tensor-product matrix.

REMARK 4.3. The Moore-Penrose inverse of a rectangular matrix of size $m \times n$ is a rectangular matrix of size $n \times m$. This can be easily verified in example 4.2 since the given matrix is a 9×9 matrix with a 9×8 nonzero block and its generalized inverse is a 9×9 matrix with a 8×9 nonzero block.

It is clear that according to Theorem 3.1, the choice of using the proposed **ginv** function depends on the fact that the tensor-product matrix T is rank deficient[4]. Hence, the proposed method can also be used for the computation of any given $m \times n$ full-rank real matrix, where $m > n$. Also, in the case when $m < n$, one can compute the transpose matrix T^* and then make use of the formula $(T^*)^\dagger = (T^\dagger)^*$. So, in order to simplify the procedure, the **ginv** function performs all necessary transpositions in all cases. Thus, the **ginv** function can also be used, directly, for the computation of any given $m \times n$ full-rank real matrix. In addition, since the user, does not know, in general, a priori whether or not the argument he/she provides is rank deficient the **ginv** function provides an explicit warning in this case.

4.2. Comparison Results. In this section, we compare the performance of the proposed method (ginv) to that of the other two algorithms, namely, the SVD method (pinv) and Courrieu's method (geninv). In addition, the accuracy of the results was examined with the matrix 2-norm in error matrices corresponding to the four properties characterizing the Moore-Penrose inverse (i.e., $TT^\dagger = (TT^\dagger)^*, T^\dagger T =$

[4]An $n \times n$ matrix T is *rank deficient* if $rank(T) = m < n$.

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

$(T^\dagger T)^*, TT^\dagger T = T, T^\dagger TT^\dagger = T^\dagger$) and the cumulative results are presented in Table 4.2.

TABLE 4.2
Error Results for rank-2^n matrices, $n = 8, 9, 10, 11, 12$.

	Rank	$\|TT^\dagger T - T\|_2$	$\|T^\dagger TT^\dagger - T^\dagger\|_2$	$\|TT^\dagger - (TT^\dagger)^*\|_2$	$\|T^\dagger T - (T^\dagger T)^*\|_2$
SVD method (Matlab pinv)	2^8	1.765×10^{-12}	1.5573×10^{-12}	7.2466×10^{-13}	6.1911×10^{-13}
Courrieu's method (geninv)		5.4709×10^{-8}	1.8213×10^{-9}	4.3151×10^{-10}	1.7632×10^{-8}
Proposed method (ginv)		1.7441×10^{-11}	1.2075×10^{-11}	1.4535×10^{-13}	7.2225×10^{-10}
SVD method (Matlab pinv)	2^9	3.4774×10^{-12}	4.7250×10^{-12}	1.7556×10^{-12}	1.2273×10^{-12}
Courrieu's method (geninv)		1.0247×10^{-6}	4.1029×10^{-8}	2.8935×10^{-9}	9.7309×10^{-8}
Proposed method (ginv)		9.6675×10^{-11}	3.0203×10^{-8}	3.9218×10^{-13}	6.4164×10^{-9}
SVD method (Matlab pinv)	2^{10}	1.5329×10^{-11}	9.9524×10^{-12}	4.6853×10^{-12}	5.3752×10^{-12}
Courrieu's method (geninv)		5.2165×10^{-6}	3.1255×10^{-7}	1.0565×10^{-8}	9.4435×10^{-7}
Proposed method (ginv)		3.3869×10^{-10}	2.3629×10^{-7}	7.2036×10^{-13}	2.7511×10^{-8}
SVD method (Matlab pinv)	2^{11}	3.7990×10^{-10}	6.0937×10^{-9}	1.5855×10^{-10}	1.2520×10^{-10}
Courrieu's method (geninv)		6.8162×10^{-2}	4.4275×10^{-3}	3.9699×10^{-5}	1.7735×10^{-3}
Proposed method (ginv)		1.7449×10^{-8}	5.3528×10^{-4}	2.5498×10^{-5}	1.9404×10^{-11}
SVD method (Matlab pinv)	2^{12}	-	-	-	-
Courrieu's method (geninv)		1.5743×10^{-5}	5.8952×10^{-6}	1.8897×10^{-7}	1.9087×10^{-9}
Proposed method (ginv)		3.9876×10^{-7}	1.219×10^{-8}	5.4433×10^{-10}	4.1422×10^{-11}

The new numerical method, based on the introduction of the **ginv** function, enables us to perform fast and accurate estimations of the generalized inverse $T^\dagger$ of a tensor-product matrix for a variety of dimensions. The tested matrices were obtained using the Matlab function **rand** and they were all rank deficient with rank 2^n for $n = 8, 9, 10, 11, 12$. Figure 4.1 shows the time efficiency curves, i.e., the rank of the tested matrix versus the computation time (in seconds). All algorithms were carefully implemented and tested in Matlab. After a deep analysis of the results in Figure 4.1 and Table 4.2 one can easily obtain the following conclusions:

5. Conclusions.

1. The **geninv** method is sensitive to numerical rounding errors, and it has been observed that it is inaccurate in the computation of generalized inverses of full-rank ill-conditioned matrices. This remark is also included in [3].

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

ELA

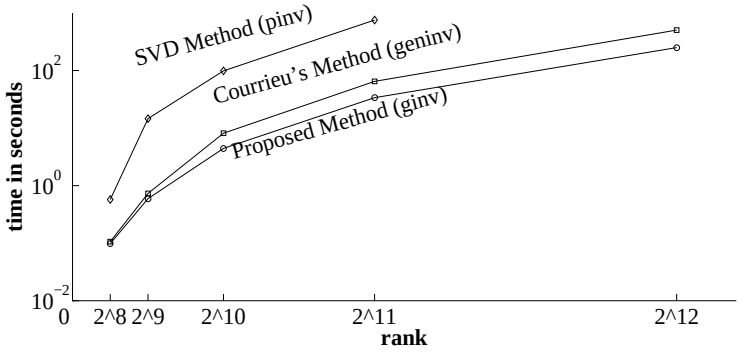

FIG. 4.1. *Time efficiency curves*

2. It is notable that for matrices with rank 2^{12} or higher, the SVD method was not able to produce a numerical result (Matlab produced an 'Out of memory' message).
3. It is evident, from Table 4.2, that using the **ginv** function we obtained a reliable approximation in all the tests that were conducted. At the same time, it is also clear that we have simplified the procedure to the extent that the interested user can reach a fast computational solution using a reduced amount of computational resources. Therefore, the proposed function allows us for a both fast and accurate computation of the Moore-Penrose inverse matrix.
4. In line 9 of the **ginv** function we included a rank test in order to simplify the use of the proposed Matlab function for the interested user. It is notable that in our tests the rank test costs more than the 50% of the computational time that the **ginv** function took to respond. Therefore, if we are in position to know more about our input data then, after a slight modification, it is clear that the **ginv** function provides a rapid method for computing generalized inverses.

6. The reverse order law. In this section, we introduce sufficient conditions so that the generalized inverse of the product of two square rank-n matrices is the product of the generalized inverses of the corresponding matrices in reverse order. In general, the well known reverse order law, $(AB)^{-1} = B^{-1}A^{-1}$ which holds for operators and matrices is only known to hold for generalized inverses under certain conditions. A lot of work has been carried out with respect to conditions so that

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

the relationship $(AB)^\dagger = B^\dagger A^\dagger$ holds (e.g.,[1] [5], [7]). The following theorem is a restatement of a part of R. Bouldin's theorem 3.1 [1] which is valid for operators and matrices.

THEOREM 6.1. *Let A, B be bounded operators with closed range on a Hilbert space $\mathcal{H}$. Then $(AB)^\dagger = B^\dagger A^\dagger$ if and only if the following three conditions hold:*

(i) The range of AB is closed,
(ii) $A^\dagger A$ commutes with BB^*,
(iii) $BB^\dagger$ commutes with A^*A.

Since in the case of matrices the range is always closed, the first condition of this theorem always holds.

PROPOSITION 6.2. *Let $T_1 = \sum_{i=1}^{n} e_i \otimes f_i$ and $T_2 = \sum_{i=1}^{n} e'_i \otimes f'_i$ be two tensor-product matrices. If $[f'_1, ..., f'_n] = [e_1, ..., e_n]$, then $(T_1T_2)^\dagger = T_2^\dagger T_1^\dagger$.*

Proof. It is well known that, $R(TT^*) = R(T)$ and $R(T^*T) = R(T^*)$. Therefore, $R(T_2T_2^*) = R(T_2) = [f'_1, ..., f'_n]$ and $R(T_1^*T_1) = R(T_1^*) = [e_1, ..., e_n]$. In order to prove condition (ii) of Theorem 6.1 we have to show that $T_1^+T_1$ commutes with the matrix $T_2T_2^*$. Indeed, since $T_1^+T_1$ is the projection matrix on $R(T_1^+) = R(T_1^*) = [e_1, ..., e_n]$, this is equivalent to $[e_1, ..., e_n]$ be invariant from $T_2T_2^*$. By the relation

$$R(T_2T_2^*) = R(T_2) = [f'_1, ..., f'_n] = [e_1, ..., e_n]$$

it is evident that condition (ii) of Theorem 6.1 holds.

In order to prove condition (iii) of Theorem 6.1 we use similar arguments as before, i.e., we must show that $T_2T_2^+$, which is the projection matrix on $R(T_2) = [f'_1, ..., f'_n]$, commutes with $T_1^*T_1$, or else that the subspace $[f'_1, ..., f'_n]$ is invariant under $T_1^*T_1$, which holds from $R(T_1^*T_1) = R(T_1^*) = [e_1, ..., e_n]$. □

By the above proposition, it is clear that only the placement of the non-zero columns plays a role on the inverse order law. Note that the law does not depend on the actual entries of the first matrix. We justify the situation with a simple example:

EXAMPLE 6.3. Let T_1 and T_2 be two tensor-product matrices of $\mathbb{R}^{5\times5}$, where $T_1 = \sum_{i=1}^{3} e_i \otimes f_i$ is a rank-3 tensor-product matrix, with

$$e_1 = (1,0,0,0,0), e_2 = (0,1,0,0,0), e_3 = (0,0,1,0,0),$$

and

$$f_1 = (1,3,5,6,7), f_2 = (2,4,3,7,6), f_3 = (1,8,7,5,6).$$

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

Then, $T_1 = \begin{pmatrix} 1 & 2 & 1 & 0 & 0 \\ 3 & 4 & 8 & 0 & 0 \\ 5 & 3 & 7 & 0 & 0 \\ 6 & 7 & 5 & 0 & 0 \\ 7 & 6 & 6 & 0 & 0 \end{pmatrix}$.

Let $T_2 = \sum_{i=1}^{3} e'_i \otimes f'_i$ be another rank-3 tensor-product matrix, with

$$e'_1 = (0,0,0,0,1), e'_2 = (0,0,0,1,0), e'_3 = (0,0,1,0,0),$$

and

$$f'_1 = (1,2,-2,0,0), f'_2 = (4,0,2,0,0), f'_3 = (0,0,-1,0,0).$$

Then, $T_2 = \begin{pmatrix} 0 & 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & -1 & 2 & -2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$.

In order to prove that $[e_1, e_2, e_3] = [f'_1, f'_2, f'_3]$, we shall use the Matlab **rref** function which produces the reduced row echelon form of the given matrix using Gauss Jordan elimination with partial pivoting. In particular, we use the following code:

```
d=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;...
1 2 -2 0 0;4 0 2 0 0;0 0 -1 0 0];
v=rref(d)
```

Then, the results are as follows:

```
v =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

which implies that $[e_1, e_2, e_3] = [f'_1, f'_2, f'_3]$.

On the other hand, using the **ginv** function it is easy to prove that

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

$$(T_1T_2)^\dagger = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{-296}{1331} & \frac{-1136}{2645} & \frac{534}{1601} & \frac{-267}{1298} & \frac{619}{2747} \\ \frac{-269}{5856} & \frac{-130}{1649} & \frac{787}{9951} & \frac{-143}{3664} & \frac{461}{8694} \\ \frac{149}{2099} & \frac{263}{4347} & \frac{-253}{2114} & \frac{2577}{26801} & \frac{-133}{4030} \end{pmatrix} = T_2^\dagger T_1^\dagger$$

7. Appendix: Matlab code of the 'ginv', 'ginvtest' functions.

The **ginv** function

```
function ginv = ginv(X)
%Returns the Moore-Penrose inverse of the argument
if isempty(X)
    quick return
 ginv = zeros(size(X'),class(X));
  return
end
[n,m]=size(X);
if rank(X) < min(n,m);
    error('matrix must be of full rank');
else
    if n > m,
C = X'*X ;
ginv = C\X';
else
    C = X*X';
    G = C\X;
    ginv = G';
    end
end
```

The **ginvtest** function

```
function Ginvtest = ginvtest(E)
%Returns the recorded errors for each one of the three
%tested methods.
format short e
    G1 = pinv(E);    %SVD method
```

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

ELA


```
    G2 = geninv(E);  %Courrie's method
    G3 = ginv(E);   %Proposed method
%The error matrices for each one of the previous methods.
    D1 = E*G1*E-E;
    D2 = G1*E*G1-G1;
    D3 = (E*G1)'-E*G1;
    D4 = (G1*E)'-G1*E;
    F1 = E*G2*E-E;
    F2 = G2*E*G2-G2;
    F3 = (E*G2)'-E*G2;
    F4 = (G2*E)'-G2*E;
    R1 = E*G3*E-E;
    R2 = G3*E*G3-G3;
    R3 = (E*G3)'-E*G3;
    R4 = (G3*E)'-G3*E;
%The accuracy of the algorithms were tested
%with the matrix 2-norm.
    Pinverror1 = norm(D1,2);
    Pinverror2 = norm(D2,2);
    Pinverror3 = norm(D3,2);
    Pinverror4 = norm(D4,2);
    Geninverror1 = norm(F1,2);
    Geninverror2 = norm(F2,2);
    Geninverror3 = norm(F3,2);
    Geninverror4 = norm(F4,2);
    Ginverror1 = norm(R1,2);
    Ginverror2 = norm(R2,2);
    Ginverror3 = norm(R3,2);
    Ginverror4 = norm(R4,2);
    Ginvtest = [Pinverror1 Pinverror2 Pinverror3...
    Pinverror4;Geninverror1 Geninverror2 Geninverror3...
    Geninverror4;Ginverror1 Ginverror2 Ginverror3 Ginverror4];
```

REFERENCES

[1] R. Bouldin. The pseudo-inverse of a product. *SIAM Journal on Applied Mathematics*, 24:(4), 489–495, 1973.

[2] A. Ben-Israel and T. N. E. Grenville. *Generalized Inverses: Theory and Applications*. Springer-Verlag, Berlin 2002.

[3] P. Courrieu. Fast Computation of Moore-Penrose Inverse matrices. *Neural Information Processing-Letters and Reviews*, 8:25–29, 2005.

[4] D. S. Djordjevic. Products of EP operators on Hilbert spaces. *Proceedings of the American*

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 17, pp. 637-650, November 2008

ELA

Mathematical Society, 129:(6), 1727–1731, 2001.
[5] T. N. E. Greville. Note on the generalized inverse of a matrix product. *SIAM Review*, 8:518–521, 1966.
[6] C. Groetsch. Generalized Inverses of Linear Operators. Marcel-Dekker, Inc., New York, 1973.
[7] S. Izumino. The product of operators with closed range and an extension of the reverse order law. *Tohoku Mathematical Journal*, 34:43–52, 1982.
[8] S. Karanasios and D. Pappas. Generalized inverses and special type operator algebras. *Facta Universitatis(NIS), Series Mathematics and Informatics*, 21:41–48, 2006.
[9] MATLAB v. 7.1, Help Browser.
[10] MATLAB User's Guide, The MathWorks Inc.
[11] M.A. Rakha. On the Moore-Penrose generalized inverse matrix. *Applied Mathematics and Computation*, 158:185-200, 2004.
[12] J. Ringrose. *Compact non self adjoint operators*. Van Nostrand London 1971.