Pivot-Free Block Matrix Inversion*

Stephen M. Watt
Ontario Research Centre for Computer Algebra
Department of Computer Science
University of Western Ontario
London Ontario, CANADA N6A 5B7
watt@csd.uwo.ca

Abstract

We present a pivot-free deterministic algorithm for the inversion of block matrices. The method is based on the Moore-Penrose inverse and is applicable over certain general classes of rings. This improves on previous methods that required at least one invertible on-diagonal block, and that otherwise required row- or column-based pivoting, disrupting the block structure. Our method is applicable to any invertible matrix and does not require any particular blocks to invertible. This is achieved at the cost of two additional specialized matrix multiplications and, in some cases, requires the inversion to be performed in an extended ring.

1. Introduction

Algorithms for block matrices have been well studied and have many benefits under certain conditions: They may be used to improve memory hierarchy performance, at the level of cache, disk or network. They may be used on a quad-tree representation that allows reasonable memory use for dense, sparse or structured matrices. They allow the generic formulation of many algorithms of linear algebra so that they may be applied over non-commutative domains. This allows some some algorithms to be expressed in a recursive formulation, leading to improved computational complexity. Block matrices have been used in numerical computing, and also in symbolic computing [1].

This paper revisits the question of block algorithms for matrix inversion. We are interested in this problem for several reasons:

First, it is an interesting mathematical problem. While many problems that are na ıvely expressed using matrix inverse (such as the solution of linear systems) are better solved using other methods (such as PLU factorization), matrix inverse nevertheless remains a fundamental operation.

Second, we wish to use block matrices as one representation in the mathematical libraries for Aldor [2]. Aldor supports generic programming through parametric polymorphism, where types must satisfy specific type categories. For matrices to satisfy the type category `Ring` they must provide a partial inverse operation.

Third, we wish to use block matrices as a test case in our work benchmarking the performance of compilers on generic programs [3].

We consider the inversion of 2×2 block matrices. Larger matrices are expressed by applying this construction recursively. Earlier methods for block matrix inversion either require one of the blocks to be invertible or degenerate into individual row or column operations, destroying the block structure.

We ask whether it is possible to formulate deterministic block matrix inversion in such a way that

1. only operations on entire blocks are used,
2. no case-based branching is required,
3. block inverses are required only when they are guaranteed to exist and
4. applied recursively, the method gives inversion the same complexity as matrix multiplication?

As with other block-oriented methods, we do not require numerical stability.

We are able to answer this question positively. We show such a block matrix inversion that is applicable over a general class of rings. In particular, it may be used for block matrices with real, complex or finite field entries.

The rest of the paper is organized as follows: Section 2 provides some necessary background. Section 3 presents our method for block matrix inverse. The algorithm is presented first for formally real rings, then for the complex and general field cases. Finally we present our conclusions.

*Accepted to 8th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 26-29 September 2006, Timişoara, Romania.

Dagstuhl Seminar Proceedings 06271
Challenges in Symbolic Computation Software
http://drops.dagstuhl.de/opus/volltexte/2006/780

2. Preliminaries

2.1 Recursive Block Matrices

We consider $n \times n$ matrices with elements from a ring R and denote the matrix ring $R^{n\times n}$. We for convenience we require that R have unity and explicitly state any additional properties when they are required. If n is even, we may put the matrices of $R^{n\times n}$ in a one-to-one correspondence with the matrices of $(R^{n/2\times n/2})^{2\times 2}$. For an $n \times n$ matrix M we take

$$M = \left[\begin{array}{cc} A & B \\ C & D \end{array} \right]$$

with the $\frac{n}{2} \times \frac{n}{2}$ matrices A, B, C, D having elements $A_{ij} = M_{i,j}$, $B_{ij} = M_{i,j+\frac{n}{2}}$, $C_{ij} = M_{i+\frac{n}{2},j}$, $D_{ij} = M_{i+\frac{n}{2},j+\frac{n}{2}}$ with $1 \leq i, j, \leq n/2$. For general n, there are two ways to impose a 2×2 block structure that preserves the multiplicative properties of the matrix ring:

The first method is to embed $R^{n\times n}$ in a ring of larger matrices $R^{(n+\ell)\times(n+\ell)}$ by adding ones along the diagonal and zeros elsewhere. If n is odd and ℓ is one, then we may apply the 2×2 block division once. If $\ell = 2^{\lceil \log_2 n \rceil} - n$ (*i.e.* if $n+\ell$ is the next power of 2), then the construction may be applied $\lceil \log_2 n \rceil$ times to obtain a fully recursive 2×2 structure. This method has the advantage that, at each level, the matrix elements are from a specific ring. For the recursive block matrix ring, we introduce the notation $R^{(2\times 2)^k}$. This allows certain isomorphisms to be expressed conveniently

$$R^{2^k\times 2^k} \cong (R^{2^{k-1}\times 2^{k-1}})^{2\times 2} \cong (R^{(2\times 2)^{k-i}})^{(2\times 2)^i} \cong R^{(2\times 2)^k}.$$

The second method is to divide matrices unevenly. The blocks A, B, C, D have elements $A_{ij} = M_{i,j}$, $B_{ij'} = M_{i,j'+\ell}$, $C_{i'j} = M_{i'+\ell,j}$, $D_{i'j'} = M_{i'+\ell,j'+\ell}$, where ℓ is between 1 and n and the indices range as $1 \leq i, j \leq \ell$ and $1 \leq i', j' \leq n - \ell$. This construction may be applied repeatedly to obtain a fully recursive 2×2 block structure. For the complexity results given below, we choose $\ell = \lfloor \frac{n}{2} \rfloor$.

Both methods may be used for the block matrix inversion problem considered in this paper. Note that in both methods the blocks A and D are square.

2.2. Naïve Block Matrix Inverse

Perhaps the most æsthetically pleasing formulation of the block matrix inverse is as

$$M^{-1} = \left[\begin{array}{cc} (A - BD^{-1}C)^{-1} & (C - DB^{-1}A)^{-1} \\ (B - AC^{-1}D)^{-1} & (D - CA^{-1}B)^{-1} \end{array} \right],$$

which is valid over any ring, and may be applied recursively.

This formulation has two defects, however: First, it requires all of A, B, C and D to be invertible. In fact, M may be invertible even if *all* of A, B, C and D are non-invertible. Second, this expression for the matrix inverse requires 8 block inversions and 8 block multiplications. When applied recursively to give an algorithm for $n \times n$ matrices, this leads to $O(n^3)$ coefficient ring operations. Since faster methods are well known and practical, this formulation is not attractive for a generic algorithm.

To avoid these problems, a more common formulation for block matrix inversion is as

$$M^{-1} = \left[\begin{array}{cc} I & -A^{-1}B \\ 0 & I \end{array} \right] \left[\begin{array}{cc} A^{-1} & 0 \\ 0 & S_A^{-1} \end{array} \right] \left[\begin{array}{cc} I & 0 \\ -CA^{-1} & I \end{array} \right]$$

$$= \left[\begin{array}{cc} A^{-1} + A^{-1}BS_A^{-1}CA^{-1} & -A^{-1}BS_A^{-1} \\ -S_A^{-1}CA^{-1} & S_A^{-1} \end{array} \right] \quad (1)$$

where $S_A = D - CA^{-1}B$ is the Schur complement of A in M. Alternatively, the inverse may be expressed as

$$M^{-1} = \left[\begin{array}{cc} I & 0 \\ -D^{-1}C & I \end{array} \right] \left[\begin{array}{cc} S_D^{-1} & 0 \\ 0 & D^{-1} \end{array} \right] \left[\begin{array}{cc} I & -BD^{-1} \\ 0 & I \end{array} \right]$$

$$= \left[\begin{array}{cc} S_D^{-1} & -S_D^{-1}BD^{-1} \\ -D^{-1}CS_D^{-1} & D^{-1} + D^{-1}CS_D^{-1}BD^{-1} \end{array} \right]. \quad (2)$$

This formulation requires only that either A or D and the corresponding Schur complement be invertible. It also has the benefit of computational efficiency, requiring only 2 inversions, 6 multiplications and 3 additive operations on blocks. If $2 \leq \omega \leq 3$ is the exponent such that $n \times n$ matrix multiplication requires $O(n^\omega)$ element operations, then this formulation of block matrix inverse, applied recursively, leads to an $O(n^\omega)$ algorithm for inversion. For concreteness, Strassen's matrix multiplication [4] requires $n^{\log_2 7}$ multiplications and $6(n^{\log_2 7} - n^2)$ additive operations $(+, -)$, and the above formulas give matrix inversion in $(6n^{\log_2 7} - 1)/5$ multiplicative operations ($\times$, inverse) and $(72n^{\log_2 7} - 165n^2 + 93n)/10$ additive operations.

2.3 Non-Invertible Blocks

While the complexity of the block matrix inversion given by equations (1) and (2) is acceptable, there remains the problem that it requires either A or D to be invertible. However, as stated earlier, all blocks may be non-invertible, even for invertible M. For example, the following permutation matrix is invertible:

$$\left[\begin{array}{cc} \left[\begin{array}{cc} 1 & 0 \\ 0 & 0 \end{array} \right] & \left[\begin{array}{cc} 0 & 0 \\ 1 & 0 \end{array} \right] \\ \left[\begin{array}{cc} 0 & 1 \\ 0 & 0 \end{array} \right] & \left[\begin{array}{cc} 0 & 0 \\ 0 & 1 \end{array} \right] \end{array} \right]$$

The usual approach to deal with this problem is to abandon the block formulation of the inverse, and use a row-oriented divide and conquer PLU matrix factorization to compute the inverse using pivots to avoid non-invertible elements. See, for example, [5].

This has two problems: The first problem is that if we are representing matrices in memory using a recursive block structure (say, as quad-trees), then a row-oriented PLU decomposition either requires converting the matrix in and out of the desired representation or requires using an indexing scheme that carries a high overhead. The second problem is that if the entries are not themselves block sub-matrices, but members of some non-commutative base ring, then there is no internal row structure to exploit.

Some authors, *e*.g. [1], argue that they do not encounter non-invertible blocks in practice. This may indeed be the case when working with specialized families of matrices or with matrices with elements chosen randomly from a large ring. For a general solution, however, non-invertibility of blocks remains a problem. For example, when the entries are chosen from a restricted set a significant proportion of small matrices can be non-invertible. Since recursive block algorithms operate upon many small matrices in their execution, non-invertible blocks may be encountered with a significant probability. For concreteness, we show below the proportion of $\mathbb{Q}^{n\times n}$ matrices that are non-invertible[1] for small n when elements are chosen from the sets $\{0, 1\}$ and $\{-1, 0, 1\}$. Note that for $n < 8$, *most* of the $\{0, 1\}$ matrices are non-invertible.

$\{0,1\}^{n\times n} \subset \mathbb{Q}^{n\times n}$

n	1	2	3	4	5	6	7	8
% non-inv.	50	62	66	66	63	58	52	45

$\{-1,0,1\}^{n\times n} \subset \mathbb{Q}^{n\times n}$

n	1	2	3	4
% non-inv.	33	41	40	45

2.4. The Moore-Penrose Inverse

The Moore-Penrose inverse is a generalized form of matrix inversion applicable to non-square matrices, discovered independently by E. Moore [7] and R. Penrose [8]. If Z is a complex $n \times m$ matrix, and Z^* its conjugate transpose, then the Moore-Penrose inverse, denoted Z^+, satisfies:

$$ZZ^+Z = Z \qquad (ZZ^+)^* = ZZ^+$$
$$Z^+ZZ^+ = Z \qquad (Z^+Z)^* = Z^+Z.$$

If Z is of full rank, then $m \le n$, Z^*Z is invertible and the Moore-Penrose inverse is

$$Z^+ = (Z^*Z)^{-1}Z^*.$$

In this case, Z^+ is left inverse of Z. If Z is square and invertible, then the Moore-Penrose inverse is Z^{-1}.

[1]These are calculated using the sequences A046747 and A056989 [6].

3. Pivot-Free Inversion

3.1 Preconditioning for Block Inversion

To apply the recursive block methods given by (1) or (2), we must find a way to guarantee that the required blocks will be invertible. We do this by preconditioning the matrix M. For invertible M, we may choose any invertible G and have

$$M^{-1} = (GM)^{-1}G. \tag{3}$$

A choice of G that guarantees invertibility of the required blocks of GM using (1), leads to a block algorithm for M^{-1}. There is the additional cost of two matrix multiplications, which can clearly be performed using block operations.

We observe that if R is embedded in a larger ring, S, and G is an invertible matrix in $S^{n\times n}$, then (3) may still be used to compute M^{-1}. The computation will use the ring operations of S, but the final result will be a matrix in $R^{n\times n}$.

For convenience, we shall assume from this point forward that R is a division ring. This guarantees that the inverses of intermediate expressions in R exist when required. If this condition is relaxed, then it may be necessary to work in some localization of R to deal with non-invertible elements that occur in intermediate expressions.

The matrix G may be chosen to randomize M to yield a probabilistic block algorithm for M^{-1}. This is often sufficient for many applications.

For a deterministic algorithm for M^{-1}, we must select a specific matrix G based on M. We show below that such a choice of G exists and can be constructed easily. In sections 3.3, 3.4 and 3.5 we show how to choose G depending on the algebraic properties of R. In each case we use Lemma 1 of section 3.2.

3.2 Two Lemmata

Lemma 1 (Block Inverse)**.** *If R is a division ring, $M \in R^{n\times n}$ is invertible and there exists $G \in E^{n\times n}$, $E \supset R$, such that all principal minors of GM are invertible, then $(GM)^{-1}$, and hence $M^{-1} = (GM)^{-1}G \in R^{n\times n}$, may be computed using only block operations. Block operations are ring operations in $E^{(2\times 2)^i}$.*

Proof. When the inversion scheme given by equation (1) is applied recursively to GM, the only blocks that are ever inverted at any level of recursion are square sub-matrices lying on the diagonal and their Schur complements with respect to their immediately containing blocks. The square sub-matrices on the diagonal are a special case of the principal minors and are therefore all invertible. Now suppose that the inverse of a Schur complement is required when inverting the containing block

$$H = \begin{bmatrix} W & X \\ Y & Z \end{bmatrix}$$

In this case we have already that H and W are invertible and we require the inverse $S_W^{-1} = (Z - YW^{-1}X)^{-1}$. The invertibility of H implies this exists, as S_W^{-1} is an explicit sub-block of H^{-1}. □

We note that the two matrix multiplications required in the inverse computation are for special forms of matrices. One multiplication is of a matrix with its own transpose, and the other is the multiplication of a symmetric matrix with a general matrix. We show that, depending on the method used for matrix multiplication, the first of these can be performed faster than a general multiplication.

Lemma 2. *For $M \in R^{n\times n}$, computing the the product M^TM requires at most*

$$\frac{n^2(2n^{\omega-2} + 2^\omega - 6)}{2^\omega - 4}$$

and requires at least $n^\omega/2^\omega$ multiplications in R.

Proof. The upper bound follows from recursive application of the the formula

$$\begin{aligned} M^TM &= \begin{bmatrix} A^T & C^T \\ B^T & D^T \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} \\ &= \begin{bmatrix} A^TA + C^TC & A^TB + C^TD \\ (A^TB + C^TD)^T & B^TB + D^TD \end{bmatrix}. \end{aligned}$$

The lower bound follows by choosing

$$M = \begin{bmatrix} X^T & Y \\ 0 & 0 \end{bmatrix}$$

and noting that M^TM computes XY. □

For the best asymptotic algorithms the upper bound from this lemma is worse than the cost of general matrix multiplication and so is not useful. When Strassen matrix multiplication is used, however, general multiplication costs $n^{\log_2 7}$ and this method requires only $1/3n^2 + 2/3n^{\log_2 7}$.

3.3 The Real Case

We begin by treating matrices over a formally real ring that need not be commutative. A ring R is *formally real* if, for any subset $\{a_i\}_{i=1,...,n} \subset R$,

$$\sum_{i=1}^{n} a_i^2 = 0 \quad \Rightarrow \quad a_1 = a_2 = \cdots = a_n = 0.$$

In particular, the rational numbers and real numbers are formally real, as are, *e.g.*, $\mathbb{Q}[\sqrt{2}]$, $\mathbb{Z}(x_1, \ldots, x_n)$ and $R[x, \partial]$ for formally real R. The complex numbers and rings of finite characteristic are not formally real.

We may now state:

Theorem 1. *If R is a formally real division ring and $M \in R^{n\times n}$ is invertible, then it is possible to compute M^{-1} as $(M^TM)^{-1}M^T$ using only block operations. By block operations, we mean ring operations in $R^{(2\times 2)^i}$.*

Proof. If M is invertible, then so are M^T and M^TM. We show that all the principal minors of M^TM are invertible, then Lemma 1 gives the required result.

Let N be a principal minor of M^TM and $S \subset \{1, ..., n\}$ the index set of its rows and columns in M^TM. Then $N = {M_S}^TM_S$, where M_S is the matrix of the columns of M indexed by S. If N were not invertible, then there would be a non-zero vector x such that $Nx = 0$. This would imply $x^TNx = (M_Sx)^T(M_Sx) = 0$. As this is a sum of squares in the formally real ring R, it would imply $M_Sx = 0$. But M is invertible and so M_S is of full column rank, which implies $M_Sx \neq 0$, a contradiction. □

The expression $(M^TM)^{-1}M^T$ is the real form of the Moore-Penrose inverse, since in this case $M^* = M^T$.

3.4 The Complex Case

We now consider the case of complex numbers in a general algebraic setting. Let C be a division ring with a formally real sub-ring R and involution "$*$", such that for all $c \in C$, $c^* \times c$ is a sum of squares in R. One such case is the complexification of a formally real ring R, that is $C = R[i]/\langle i^2 + 1\rangle$ with complex conjugation as the involution: $(a + bi)^* = a - ib$ for $a, b \in R$. A second case is to take C as the quaternions over R with the involution $(a + bi + cj + dk)^* = a - bi - cj - dk$ for $a, b, c, d \in R$.

We may lift the involution on C to one on $C^{n\times n}$ by defining M^* as the transpose of the element-wise involution of M. Now the arguments presented in section 3.3 all follow when M^T is replaced by M^*. We may thus compute M^{-1} with block operations using the Moore-Penrose inverse:

Theorem 2. *Let C be a division ring with a formally real sub-ring R and involution "$*$", such that for all $c \in C$, $c^* \times c$ is a sum of squares in R. If $M \in C^{n\times n}$ is invertible, then it is possible to compute M^{-1} as $(M^*M)^{-1}M^*$ using only block operations. Here, block operations are ring operations in $C^{(2\times 2)^i}$.*

Proof. Replace x^T and M^T with x^* and M^* in the proof of Theorem 1 to obtain the same contradiction. □

3.5 General Fields

For rings that do not have a formally real sub-ring, the approach of the previous sections may not be applied directly. We can, however, work in a convenient extended ring.

We use an idea of [9] and [10] who work in a rational function field. Mulmuley [9] has approached the problem of fast computation of matrix rank over an arbitrary field, K, by working in the field of univariate rational functions $K(t)$. Diaz-Toca *et al* [10] have extended this approach to generalize Cramer's rule. They introduce a generalized form of the Moore-Penrose inverse, which in our setting gives

$$M^{-1} = (M^\circ M)^{-1} M^\circ$$
$$M^\circ = Q_n^{-1} M^T Q_n$$

where $Q_n = \mathrm{diag}(1, t, t^2, \ldots, t^{n-1})$.

We find this formulation of matrix inverse to be suitable for pivot-free block matrix inversion in $K^{n\times n}$.

Theorem 3. *Let K be a field. If $M \in K^{n\times n}$ is invertible, then it is possible to compute M^{-1} as $(M^\circ M)^{-1}M^\circ$ using only block operations. Here, block operations are ring operations in $K(t)^{(2\times 2)^i}$.*

Proof. We show that all principal minors of $M^\circ M$ are invertible. Let N be a $s \times s$ principal minor of $M^\circ M$ and $S \subset \{1, ..., n\}$, the index set of its rows and columns in $M^\circ M$. Let $P_S \in \{0,1\}^{n\times s} \subset K(t)^{n\times s}$ be the projection matrix that, when multiplying an $n \times n$ matrix on the right, retains the columns indexed by S. Then we have

$$N = P_S^T ({Q_n}^{-1} M^T Q_n M) P_S. \quad (4)$$

If N is non-invertible, then there is a non-zero vector $x \in K(t)^s$ such that $Nx = 0$. In that case we also have

$$(x^T P_S^T Q_n P_S)\, Nx = 0. \quad (5)$$

Combining (4) and (5) gives

$$\begin{aligned}
&(x^T P_S^T Q_n P_S)\, Nx \\
&\qquad = (x^T P_S^T Q_n P_S)\;\; P_S^T ({Q_n}^{-1} M^T Q_n M P_S)\, x \\
&\qquad = (M P_S\, x)^T Q_n (M P_S\, x) \\
&\qquad = \sum_{i=1}^{n} (M P_S\, x)_i{}^2\, t^{i-1} = 0. \qquad (6)
\end{aligned}$$

But M is invertible by hypothesis and the vector $P_S x$ is non-zero. Therefore $M P_S\, x \neq 0$ and the polynomial (6) cannot vanish, which is a contradiction. □

4. Conclusions

We have shown how the Moore-Penrose inverse can be used to give a formulation of recursive block matrix inversion that rules out non-invertible sub-blocks. This is useful for a wide range of element rings.

This formulation requires two additional matrix multiplications at the top level only. A software implementation may choose to use the usual, somewhat less expensive, formulation of block matrix inversion given by equations (1) and (2) and resort to this method only if it encounters a step where both A and D are non-invertible.

Finally we note that, because our formulation guarantees the required blocks will be invertible, it does not require special cases and may therefore be more suitable for a hardware implementation.

Acknowledgements

The author would like to thank Wayne Eberly, Laureano Gonzalez Vega, David Jeffrey, Victoria Powers, B. David Saunders and Arne Storjohann for helpful discussions, many of which occurred during the 2006 Dagstuhl workshop *Challenges in Symbolic Computation.*

References

[1] S.K. Abdali and D.S. Wise: *Experiments with quadtree representation of matrices.* In P. Gianni (ed.), Proc. ISSAC 88, LNCS 358, Springer Verlag (1989), 96–108.

[2] S.M. Watt: *Aldor.* In Handbook of Computer Algebra, J. Grabmeier, E. Kaltofen and V. Weispfenning (editors), Springer Verlag (2003), 265–270.

[3] L. Dragan and S.M. Watt: *Performance Analysis of Generics for Scientific Computing.* Proc. 7th Int'l Symposium on Symbolic and Numeric Algorithms in Scientific Computing, (SYNASC 2005), September 25-29 2005, IEEE Press, 93-100.

[4] V. Strassen: *Gaussian Elimination is Not Optimal*, Numerische Mathematik 13, 354-356 (1969).

[5] A. Aho, J. Hopcroft, and J. Ullman: *The Design and Analysis of Computer Algorithms.* Addison-Wesley, 1974.

[6] N.J.A. Sloane: *The On-Line Encyclopedia of Integer Sequences*, `http://www.research.att.com/~njas/sequences/` (Valid September 2006).

[7] E. H. Moore: *On the Reciprocal of the General Algebraic Matrix.* Bulletin of the AMS 26, 394-395 (1920).

[8] R. Penrose: *A Generalized Inverse for Matrices.* Proc. of the Cambridge Philosophical Society 51, 406-413 (1955).

[9] K. Mulmuley: *A Fast Parallel Algorithm to Compute the Rank of a Matrix over an Arbitrary Field*, Combinatorica 7, 101-104 (1987).

[10] G.M. Diaz-Toca, L. Gonzalez-Vega and H. Lombardi: *Generalizing Cramer's Rule: Solving Uniformly Linear Systems of Equations*, SIAM J. Matrix Anal. Appl. 27(3), 621-637 (2005).