An efficient, block-by-block algorithm for inverting a block tridiagonal, nearly block Toeplitz matrix

COMPUTATIONAL SCIENCE&DISCOVERY

An efficient, block-by-block algorithm for inverting a block tridiagonal, nearly block Toeplitz matrix

Matthew G Reuter and Judith C Hill

Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
E-mail: reutermg@ornl.gov

Received 30 December 2011, in final form 13 June 2012
Published 31 July 2012
Computational Science & Discovery **5** (2012) 014009 (8pp)
doi:10.1088/1749-4699/5/1/014009

Abstract. We present an algorithm for computing any block of the inverse of a block tridiagonal, nearly block Toeplitz matrix (defined as a block tridiagonal matrix with a small number of deviations from the purely block Toeplitz structure). By exploiting both the block tridiagonal and the nearly block Toeplitz structures, this method scales independently of the total number of blocks in the matrix and linearly with the number of deviations. Numerical studies demonstrate this scaling and the advantages of our method over alternatives.

Contents

1. Introduction

The direct inversion of matrices is useful for countless applications, most notably for solving linear systems of equations. Unfortunately, general algorithms for this task scale poorly with the matrix size, prompting the use of approximate, iterative solvers instead [1, 2]. Algebraic structure in the matrix to be inverted can, when exploited, produce more efficient direct inversion algorithms, as seen for tridiagonal

1749-4699/12/014009+08$33.00

[3, 4], tridiagonal/Toeplitz [5], block tridiagonal [4, 6–12], block Toeplitz [13] and block tridiagonal/block Toeplitz [14, 15] matrices. Moreover, some applications, such as the electronic structure of materials [15–18], require only specific elements of the inverse matrix (as opposed to the entire matrix), resulting in additional efficiency [10, 15, 19].

Block tridiagonal and *nearly* block Toeplitz matrices, i.e. block tridiagonal, block Toeplitz matrices with a small number of deviations from the purely block Toeplitz structure (hereafter called deviations), have recently appeared when investigating surface effects in materials [15, 18]. Physically, these studies examined the persistence of surface effects into the material and, mathematically, confined deviations to the surfaces (upper left and/or bottom right matrix corners). Moreover, a small number of blocks of the inverse matrix were required in these studies—those at or near a surface—and were obtained using either (i) a block tridiagonal matrix inversion method (ignoring the nearly block Toeplitz structure) [18] or (ii) a specially designed block tridiagonal/block Toeplitz matrix inversion method where the deviations were assumed to be in particular locations (the matrix corners) [15]. The former method, while general, scales linearly with the number of blocks in the matrix, whereas the latter trades generality for constant scaling [O(1)].

Accordingly, the goal of this work is to describe a computational algorithm for calculating specific blocks (a 'block-by-block' algorithm) of the inverse of a block tridiagonal, nearly block Toeplitz matrix with a certain number of deviations in *arbitrary* locations. Such an algorithm will facilitate investigations into the effects of defects and disorder in materials. In brief, our method combines the strengths of the two previous algorithms, using the generality of the block tridiagonal method to incorporate deviations and the block tridiagonal/block Toeplitz method to accelerate computation away from any deviations. As will be demonstrated, this yields an algorithm that scales linearly with the number of deviations and independently of the total number of blocks in the matrix.

The structure of this paper is as follows. Section 2 develops our algorithm by first introducing the block tridiagonal matrix inversion algorithms (section 2.1) and then incorporating the block Toeplitz and nearly block Toeplitz structures in sections 2.2 and 2.3, respectively. We proceed to comparing our method to that for block tridiagonal matrices [10] with several numerical tests in section 3. Finally, we conclude in section 4.

2. The inversion algorithm

Mathematically, we seek blocks of $\mathbf{M}^{-1}$ where

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{C} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{B} & \mathbf{A} & \mathbf{C} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{B}' & \mathbf{A} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A} & \mathbf{C} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{B} & \mathbf{A}' & \mathbf{C} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{B} & \mathbf{A} & \mathbf{C}' & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{A} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{1}$$

is a block tridiagonal, nearly block Toeplitz matrix. In the nearly block Toeplitz structure shown in (1), blocks **A**, **B** and **C** are repeated along their diagonals with some deviations (denoted by primes) present arbitrarily along these diagonals.

In this section, we describe our method for obtaining an arbitrary block of $\mathbf{M}^{-1}$. We begin by reviewing block-by-block algorithms for inverting a block tridiagonal matrix in section 2.1. We then introduce a matrix Möbius transformation [19] and discuss its role in exploiting the block Toeplitz structure in section 2.2. Finally, we complete our algorithm in section 2.3 by incorporating deviations.

2.1. Inverting a block tridiagonal matrix

Block-by-block algorithms for inverting block tridiagonal matrices have been discussed previously [8, 10, 15]; we summarize these algorithms here. Suppose that

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{C}_2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{B}_1 & \mathbf{A}_2 & \mathbf{C}_3 & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_2 & \mathbf{A}_3 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_N \end{bmatrix} \tag{2}$$

is a general block tridiagonal matrix with N blocks along the diagonal. We also assume that the diagonal blocks are square matrices and that $\mathbf{A}_1$ and $\mathbf{A}_N$ are invertible. Then, the diagonal blocks of $\mathbf{M}^{-1}$ are

$$\left(\mathbf{M}^{-1}\right)_{n,n} = \left[\mathbf{A}_n - \mathbf{X}_n - \mathbf{Y}_n\right]^{-1}, \tag{3}$$

where

$$\mathbf{X}_n = \begin{cases} \mathbf{0} & \text{if } n = N, \\ \mathbf{C}_{n+1}\left[\mathbf{A}_{n+1} - \mathbf{X}_{n+1}\right]^{-1}\mathbf{B}_n & \text{if } 1 \leqslant n < N \end{cases} \tag{4}$$

and

$$\mathbf{Y}_n = \begin{cases} \mathbf{0} & \text{if } n = 1, \\ \mathbf{B}_{n-1}\left[\mathbf{A}_{n-1} - \mathbf{Y}_{n-1}\right]^{-1}\mathbf{C}_n & \text{if } 1 < n \leqslant N. \end{cases} \tag{5}$$

A close examination of (3)–(5) reveals that $\mathbf{A}_n - \mathbf{X}_n - \mathbf{Y}_n$ is the Schur complement of the partition of $\mathbf{M}$ excluding row/column n [20]. Furthermore, the off-diagonal blocks are

$$\left(\mathbf{M}^{-1}\right)_{m,n} = \begin{cases} -\left[\mathbf{A}_m - \mathbf{X}_m\right]^{-1}\mathbf{B}_{m-1}\left(\mathbf{M}^{-1}\right)_{m-1,n} & \text{if } m > n, \\ -\left[\mathbf{A}_m - \mathbf{Y}_m\right]^{-1}\mathbf{C}_{m+1}\left(\mathbf{M}^{-1}\right)_{m+1,n} & \text{if } m < n. \end{cases} \tag{6}$$

From (3) and (6), the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences provide all the necessary quantities for computing an arbitrary block of $\mathbf{M}^{-1}$. Moreover, only parts of the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences are needed for calculating $\left(\mathbf{M}^{-1}\right)_{m,n}$; specifically, $\mathbf{X}_{n+k}$ for $k = 0, \ldots, \max(0, m-n)$ and $\mathbf{Y}_{n-k}$ for $k = 0, \ldots, \max(0, n-m)$. Unfortunately, (4) and (5) require the recursive calculation of the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences, respectively, meaning that more sequence terms must be calculated than are actually needed. Even though the computations scale linearly with n, each step in the recurrences can be expensive, especially if the blocks of $\mathbf{M}$ are large.

2.2. Exploiting the block Toeplitz structure

Let us now assume that $\mathbf{M}$ is also block Toeplitz, that is, $\mathbf{A}_n = \mathbf{A}$, $\mathbf{B}_n = \mathbf{B}$ and $\mathbf{C}_n = \mathbf{C}$ for all appropriate n. Incorporating this structure into the recursive $\mathbf{X}_n$ sequence (4),

$$\mathbf{X}_n = \mathbf{C}\left(\mathbf{A} - \mathbf{X}_{n+1}\right)^{-1}\mathbf{B}; \tag{7}$$

the process of calculating $\mathbf{X}_n$ from $\mathbf{X}_{n+1}$ is exactly the same as calculating $\mathbf{X}_{n+1}$ from $\mathbf{X}_{n+2}$, etc. We now show that the *matrix Möbius transformation* enables exploitation of this block Toeplitz structure by combining multiple iterations of (7) into a single step.

The matrix Möbius transformation [19] is a generalization to matrices of the Möbius transformation from complex variables, which has found numerous uses in, e.g., conformal mappings and geometry [21]. Given $M \times M$ matrices $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$ and $\mathbf{z}$, the matrix Möbius transformation of $\mathbf{z}$ by $\mathbf{T}$ is defined as

$$\mathbf{T} \bullet \mathbf{z} \equiv (\mathbf{a}\mathbf{z} + \mathbf{b})(\mathbf{c}\mathbf{z} + \mathbf{d})^{-1}, \tag{8}$$

where

$$\mathbf{T} = \begin{bmatrix} \mathbf{a} & \mathbf{b} \\ \mathbf{c} & \mathbf{d} \end{bmatrix}$$

is a $2M \times 2M$ matrix. Furthermore, it is easily proven that the matrix Möbius transformation retains the property of associativity from its complex variables analogue, i.e.

$$\mathbf{S} \bullet (\mathbf{T} \bullet \mathbf{z}) = (\mathbf{ST}) \bullet \mathbf{z}, \tag{9}$$

where **S** and **T** are matrix Möbius transformations and **ST** is the standard matrix product.

Now, returning to the $\mathbf{X}_n$ sequence in (7),

$$\begin{aligned} \mathbf{X}_n &= \mathbf{C}\left[\mathbf{B}^{-1}\left(\mathbf{A} - \mathbf{X}_{n+1}\right)\right]^{-1} \\ &= \begin{bmatrix} \mathbf{0} & \mathbf{C} \\ -\mathbf{B}^{-1} & \mathbf{B}^{-1}\mathbf{A} \end{bmatrix} \bullet \mathbf{X}_{n+1} \\ &= \begin{bmatrix} \mathbf{0} & \mathbf{C} \\ -\mathbf{B}^{-1} & \mathbf{B}^{-1}\mathbf{A} \end{bmatrix}^{N-n} \bullet \mathbf{X}_N. \end{aligned} \tag{10}$$

Assuming **B** is invertible, the matrix Möbius transformation accelerates the calculation of an arbitrary $\mathbf{X}_n$; we simply need to assemble the matrix Möbius transformation in (10), diagonalize it and apply the desired power. Note that, if **B** is numerically singular, the authors of [15, 22] suggest using the singular value decomposition of **B** to construct an invertible approximation of **B** for use in (10). Similarly for the $\mathbf{Y}_n$ sequence (5),

$$\mathbf{Y}_n = \begin{bmatrix} \mathbf{0} & \mathbf{B} \\ -\mathbf{C}^{-1} & \mathbf{C}^{-1}\mathbf{A} \end{bmatrix}^{n-1} \bullet \mathbf{Y}_1. \tag{11}$$

Thus, exploitation of the block Toeplitz structure yields a constant-scaling (O(1)) algorithm for directly calculating terms in the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences. This is a considerable improvement for obtaining blocks of $\mathbf{M}^{-1}$ with (3) and (6) since we no longer have to spend time and resources computing unnecessary terms in the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences.

2.3. Incorporating deviations

Finally, suppose that there is a deviation around row/column j in the matrix ($1 \leqslant j \leqslant N$),

$$\mathbf{M} = \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \cdots \\ \cdots & \mathbf{A} & \mathbf{C}_j & \mathbf{0} & \cdots \\ \cdots & \mathbf{B}_{j-1} & \mathbf{A}_j & \mathbf{C}_{j+1} & \cdots \\ \cdots & \mathbf{0} & \mathbf{B}_j & \mathbf{A} & \cdots \\ \cdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{12}$$

such that **M** is now nearly block Toeplitz. As before, we must calculate the desired terms in the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences to use in (3) and (6). The calculations of $\mathbf{X}_n$ for $n > j$ are straightforward using (10). Computing $\mathbf{X}_{j+1}$ in that manner, (4) is used to obtain $\mathbf{X}_j$ and then $\mathbf{X}_{j-1}$,

$$\begin{aligned} \mathbf{X}_j &= \mathbf{C}_{j+1}\left[\mathbf{A} - \mathbf{X}_{j+1}\right]^{-1}\mathbf{B}_j, \\ \mathbf{X}_{j-1} &= \mathbf{C}_j\left[\mathbf{A}_j - \mathbf{X}_j\right]^{-1}\mathbf{B}_{j-1}. \end{aligned}$$

Then, for $n < j - 1$,

$$\mathbf{X}_n = \begin{bmatrix} \mathbf{0} & \mathbf{C} \\ -\mathbf{B}^{-1} & \mathbf{B}^{-1}\mathbf{A} \end{bmatrix}^{j-1-n} \bullet \mathbf{X}_{j-1}.$$

Table 1. Scaling rules for the recursive method (equations (4) and (5)) and our matrix Möbius transformation method (MTM, section 2.3). N is the number of blocks in the matrix and N_D is the number of deviations from block Toeplitz structure.

	Recursive	MTM
N	$O(N)$	$O(1)$
N_D	$O(1)$	$O(N_D)$

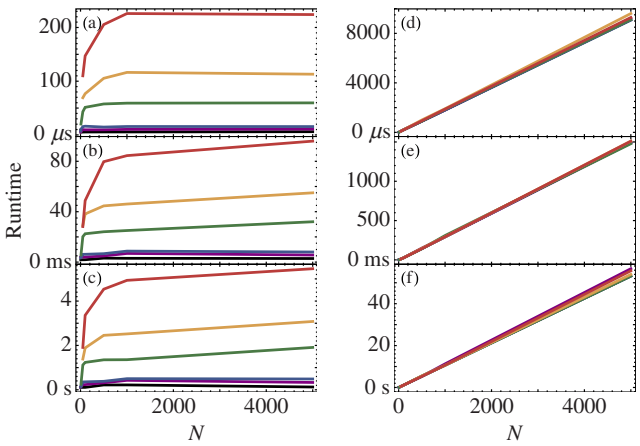

Figure 1. Average runtimes of 20 trials for calculating $\mathbf{X}_1$ and $\mathbf{Y}_N$ with the matrix MTM (left) and the recursion relations (right). Top: 1×1 block sizes, runtimes in μs; middle: 25×25, runtimes in ms; bottom: 100×100, runtimes in s. Each line represents a fixed number of deviations (N_D): 0 (black), 1 (purple), 2 (blue), 10 (green), 20 (yellow) and 40 (red). The recurrence relations scale linearly with the number of blocks (N). With 1×1 blocks, the MTM scales independently of N (for N sufficiently large), as theoretically expected. Larger block sizes, however, introduce computational effects (perhaps related to caching) with a weak dependence on N. These runtimes indicate that the MTM is advantageous for the recurrence relations for large N.

Abstractly, we use the matrix Möbius transformation (10) to skip past rows and columns that conform to the block Toeplitz structure and the recursive formulation (4) near any deviations. In this way, we compute the desired terms in the $\mathbf{X}_n$ sequence in constant time with n and in linear time with the number of deviations from the block Toeplitz structure. A similar idea is used for calculating the $\mathbf{Y}_n$ sequence. Finally, having all the required $\mathbf{X}_n$ and $\mathbf{Y}_n$ terms, the blocks of $\mathbf{M}^{-1}$ are computed with (3) and (6).

3. Computational tests

As discussed in section 2, our method for computing an arbitrary block of $\mathbf{M}^{-1}$ reduces to calculating terms in the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences. Without block Toeplitz structure, recurrence relations (4) and (5) are used, producing an algorithm that scales linearly with the number of blocks and is unfazed by deviations. On the other hand, the matrix Möbius transformation accelerates calculation of the sequences in the absence of deviations, ultimately producing an algorithm that scales constantly with the total number of matrix blocks and linearly with the number of deviations. Here we numerically compare these two techniques and substantiate the theoretical scaling rules summarized in table 1.

Code for both routines was written in C++ and serially executed on a 2.3 GHz processor; no effort was made to optimize the code for either algorithm. Since the only difference between the methods lies in calculating the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences, we compare the time needed for either method to compute both $\mathbf{X}_1$ and $\mathbf{Y}_N$ (the most expensive terms in each sequence). Figures 1 and 2 show average runtimes for both methods over a range of block sizes (M), the number of blocks (N) and the number of deviations (N_D). In particular, figure 1 verifies that the matrix Möbius transformation method (MTM) scales independently of N (figure 1(a)) and that

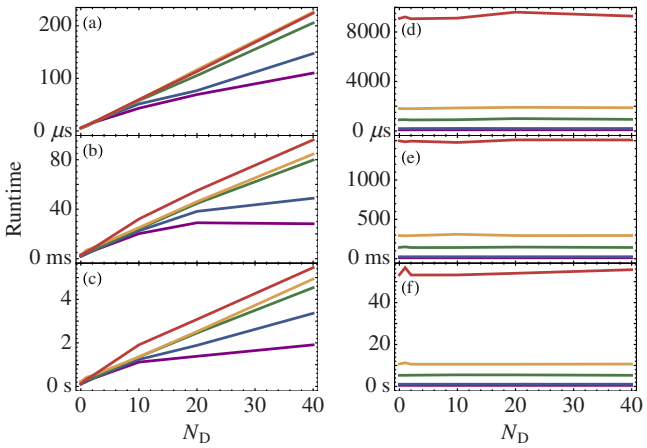

Figure 2. Average runtimes of 20 trials for calculating $\mathbf{X}_1$ and $\mathbf{Y}_N$ with the matrix MTM (left) and the recursion relations (right). Top: 1×1 block sizes, runtimes in μs; middle: 25×25, runtimes in ms; bottom: 100×100, runtimes in s. Each line represents a fixed number of blocks (N): 50 (purple), 100 (blue), 500 (green), 1000 (yellow) and 5000 (red). As expected, the MTM scales linearly with N_D and the recurrence relations scale independently of N_D.

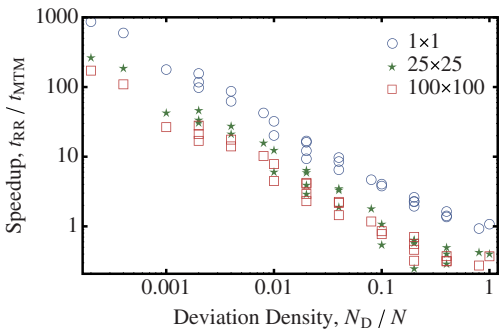

Figure 3. Speedup from using the matrix MTM over the recurrence relations (RRs) in calculating $\mathbf{X}_1$ and $\mathbf{Y}_N$ as a function of the deviation density N_D/N (N_D is the number of deviations, N is the total number of blocks, t_{RR} is the average runtime for the RRs and t_{MTM} is the average runtime for the MTM). Various block sizes (1×1, 25×25 and 100×100) all show the same trend: the MTM greatly accelerates computation of the $\mathbf{X}_n$ and $\mathbf{Y}_n$ sequences when $N_D \ll N$ and remains computationally advantageous for $N_D \lesssim N/10$.

the recursive method scales linearly with N. We note, however, that the MTM exhibits a weak dependence on N as the block size increases; this may be caused by caching effects. Similarly, figure 2 displays the expected scaling behavior with respect to N_D.

Figure 3 additionally shows the average speedup gained by the MTM relative to the recursive method as a function of the deviation density (N_D/N). Unsurprisingly, the highest speedups are realized when the deviation density is very small ($N_D \ll N$); in these cases the MTM skips past the many $\mathbf{X}_n$ and $\mathbf{Y}_n$ terms far from a deviation. Furthermore, figure 3 shows that the MTM continues to outperform the recursive method for $N_D \lesssim 0.1N$. Matrices with many deviations stretch the notion of 'nearly block Toeplitz' and do not benefit from such a structure.

4. Conclusions

Block tridiagonal, nearly block Toeplitz matrices have recently appeared in studies of materials [15, 18], and their inverses, in whole or in part, contain desirable information. In this work, we have detailed an efficient (constant-scaling with the size of the matrix and linear-scaling with the number of deviations from a purely

block Toeplitz structure) algorithm for calculating any block of the inverse of a block tridiagonal, nearly block Toeplitz matrix. Previous algorithms for this task either ignored the nearly block Toeplitz structure or assumed the deviations to be in specific locations. Our method remedies both limitations by using a matrix Möbius transformation to accelerate computation away from deviations and the recurrence relations to incorporate deviations. Owing to this favorable scaling, our method will facilitate investigations of disorder in materials, which have been computationally difficult due to necessarily large system sizes [23, 24]. Finally, although this method was designed with the study of materials in mind, we hope it is useful in other applications. For instance, this method may find use in computing preconditioners for implicit finite difference methods.

Acknowledgments

We are grateful to Rebecca Hartman-Baker, Mark Ratner and Bobby Sumpter for helpful conversations. MGR performed this research as a Eugene P Wigner Fellow and staff member of the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC, for the US Department of Energy under contract no DE-AC05-00OR22725. Congratulations to the US Department of Energy Computational Science Graduate Fellowship Program for 20 years of celebrating and cultivating excellence in computational science.

References

[1] Heath M T 2002 *Scientific Computing: An Introductory Survey* 2nd edn (New York: McGraw-Hill)
[2] Saad Y 2003 *Iterative Methods for Sparse Linear Systems* SIAM 2nd edn (Philadelphia, PA: Saunders)
[3] Thomas L H 1949 Elliptic problems in linear difference equations over a network *Technical Report* Watson Scientific Computing Laboratory, Columbia University
[4] Meurant G 1992 A review on the inverse of symmetric tridiagonal and block tridiagonal matrices *SIAM J. Matrix Anal. Appl.* **13** 707–28
[5] Evans D J 1980 On the solution of certain Toeplitz tridiagonal linear systems *SIAM J. Numer. Anal.* **17** 675–80
[6] Meyer H I and Hollingsworth B J 1957 A method of inverting large matrices of special form *Math. Tables Aids Comput.* **11** 94–7
[7] Cash J R 1982 On the solution of block tridiagonal systems of linear algebraic equations having a special structure *SIAM J. Numer. Anal.* **19** 1220–32
[8] Godfrin E M 1991 A method to compute the inverse of an *n*-block tridiagonal quasi-Hermitian matrix *J. Phys.: Condens. Matter* **3** 7843–8
[9] Ran R-S and Huang T-Z 2006 The inverses of block tridiagonal matrices *Appl. Math. Comput.* **179** 243–7
[10] Hod O, Peralta J E and Scuseria G E 2006 First-principles electronic transport calculations in finite elongated systems: a divide and conquer approach *J. Chem. Phys.* **125** 114704
[11] Petersen D E, Sørensen H H B, Hansen P C, Skelboe S and Stokbro K 2008 Block tridiagonal matrix inversion and fast transmission calculations *J. Comput. Phys.* **227** 3174–90
[12] Hirshman S P, Perumalla K S, Lynch V E and Sanchez R 2010 BCYCLIC: a parallel block tridiagonal matrix cyclic solver *J. Comput. Phys.* **229** 6392–404
[13] Labahn G, Choi D K and Cabay S 1990 The inverses of block Hankel and block Toeplitz matrices *SIAM J. Comput.* **19** 98–123
[14] Minchev B S 2003 Some algorithms for solving special tridiagonal block Toeplitz linear systems *J. Comput. Appl. Math.* **156** 179–200
[15] Reuter M G, Seideman T and Ratner M A 2011 Probing the surface-to-bulk transition: a closed-form constant-scaling algorithm for computing subsurface Green functions *Phys. Rev.* B **83** 085412
[16] Lee D H and Joannopoulos J D 1981 Simple scheme for surface-band calculations: I. *Phys. Rev.* B **23** 4988–96
[17] Lee D H and Joannopoulos J D 1981 Simple scheme for surface-band calculations: II. The Green's function *Phys. Rev.* B **23** 4997–5004
[18] Hod O, Peralta J E and Scuseria G E 2007 Edge effects in finite elongated graphene nanoribbons *Phys. Rev.* B **76** 233401
[19] Umerski A 1997 Closed form solutions to surface Green's functions *Phys. Rev.* B **55** 5266–75
[20] Zhang F 2005 *The Schur Complement and its Applications* (New York: Springer)

[21] Needham T 2005 *Visual Complex Analysis* (Oxford: Oxford University Press)
[22] Rungger I and Sanvito S 2008 Algorithm for the construction of self-energies for electronic transport calculations based on singularity elimination and singular value decomposition *Phys. Rev.* B **78** 035407
[23] Makov G and Payne M C 1995 Periodic boundary conditions in *ab initio* calculations *Phys. Rev.* B **51** 4014–22
[24] Martínez J I, Cabria I, López M J and Alonso J A 2009 Adsorption of lithium on finite graphitic clusters *J. Phys. Chem.* C **113** 939–41