INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN BIOMEDICAL ENGINEERING
Int. J. Numer. Meth. Biomed. Engng. 2010; **26**:752–759
Published online 22 September 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cnm.1172
COMMUNICATIONS IN NUMERICAL METHODS IN ENGINEERING

Variant of the Thomas Algorithm for opposite-bordered tridiagonal systems of equations

Alexandre Martin[*, †] and Iain D. Boyd

Department of Aerospace Engineering, The University of Michigan, Ann Arbor, MI 48109, U.S.A.

SUMMARY

To solve tridiagonal systems of linear equations, the Thomas Algorithm is a much more efficient method than, for instance, Gaussian elimination. The algorithm uses a series of *elementary row operations* and can solve a system of n equations in $\mathcal{O}(n)$ operations, instead of $\mathcal{O}(n^3)$. Many variations of the Thomas Algorithm have been developed over the years to solve very specific near-tridiagonal matrix. However, none of these methods address the situation of a system of linear equations that could easily be solved if elementary operations on columns are applied, instead of elementary operations on rows. The present paper proposes an efficient method that allows the use of *elementary column operations* to solve linear systems of equations using vector multiplication techniques, such as the one proposed by Thomas. Copyright © 2008 John Wiley & Sons, Ltd.

Received 23 December 2007; Revised 21 July 2008; Accepted 23 July 2008

KEY WORDS: linear solver; Thomas Algorithm; elementary column operations

1. INTRODUCTION

The idea behind Gaussian elimination is that, through elementary row operations, a given linear system of equations can be reduced to a simple form (triangular, row canonical, row echelon, etc.), which can then be solved easily through a back-substitution. The problem with that method, however, is that the number of steps required to perform a Gaussian elimination is of the order of $\mathcal{O}(n^3)$. In 1949, L. H. Thomas proposed a simplified form of the elimination process to solve tridiagonal systems [1]. This algorithm, known as the Thomas Algorithm or the Tridiagonal Matrix Algorithm, takes advantage of the fact that a tridiagonal system of linear equations can be reduced to four vectors of size n. The algorithm uses a series of elementary row operations on the vector form of the system and is able to solve it in $\mathcal{O}(n)$ operations. The algorithm can also be expanded to diagonally dominated systems (band matrices), with substantial reductions in the number of operations when compared with a regular Gaussian elimination. Since diagonal linear systems of equations occur frequently, this family of algorithms is very useful.

One of the principles of Gaussian elimination is that, by using elementary row operations, the solution vector of the system is not modified; the operations affect only the ‘augmented matrix’.

[*]Correspondence to: Alexandre Martin, Department of Aerospace Engineering, The University of Michigan, 1320 Beal Avenue, Ann Arbor, MI 48109, U.S.A.
[†]E-mail: almar@umich.edu

Contract/grant sponsor: Government of Québec; Fonds de recherche sur la nature et les technologies (NATEQ)
Contract/grant sponsor: NASA Constellation University Institutes Program; contract/grant number: NCC3-989

Algebraically, elementary row operations are achieved by applying an elementary matrix to the left side of a vector or matrix. For a linear system of equations:

$$\mathbf{Ax}=\mathbf{d} \tag{1}$$

the row multiplying matrix **S** is applied to both sides of the equation, without modifying **x**:

$$\mathbf{SAx}=\mathbf{Sd}$$

Hence, if matrix **SA** is easier to invert than matrix **A**, **x** will be easier to obtain. That principle is the whole basis of the Gaussian–Jordan elimination; elementary multiplying matrices are applied to both sides of the equation until the left side transforms into the identity matrix. In the case of the Thomas Algorithm, the matrices are applied to eliminate the superdiagonal (or supradiagonal) elements.

Many variations of the Thomas Algorithm have surfaced over the years [2, 3] with the purpose of solving very specific near-tridiagonal matrices. However, none of these methods addresses the problem of a system of linear equations that could easily be solved if elementary column operations instead of elementary row operations could be applied to matrix **A**. In this respect, this paper proposes an efficient method that allows the use of elementary column operation to solve linear systems of equations, using vector multiplication techniques such as the one proposed by Thomas [1]. First, the algebraic formulation is presented, then the solution is applied to a simple system. Finally, the algorithm is compared with another algorithm, the Sherman–Morrison–Woodbury algorithm [4].

2. ALGEBRAIC FORMULATION

In matrix notation, elementary column operations are represented by applying an elementary matrix to the right side of a given matrix. For a linear system of equations, in order to preserve the relations, the inverse of the elementary matrix must be applied to the right side of the solution vector. If **R** represents an elementary column-multiplying matrix, then Equation (1) becomes

$$\mathbf{ARR}^{-1}\mathbf{x}=\mathbf{d}$$

As is the case for a regular Gaussian elimination (or the Thomas Algorithm), a succession of **R**-type matrices are applied to matrix **A** so that the transformed matrix is easier to invert:

$$\mathbf{AR}_1\ldots\mathbf{R_n}\mathbf{R_n}^{-1}\ldots\mathbf{R}_1^{-1}\mathbf{x}=\mathbf{d} \tag{2}$$

By defining $\mathbf{y}=\mathbf{R_n}^{-1}\ldots\mathbf{R}_1^{-1}\mathbf{x}$ and $\mathbf{\Lambda}=\mathbf{AR}_1\ldots\mathbf{R_n}$, the new system of equations is as follows:

$$\mathbf{\Lambda y}=\mathbf{d}$$

Unlike the Gaussian elimination (or the Thomas Algorithm), the inversion of matrix $\mathbf{\Lambda}$ will not lead to the straightforward evaluation of **x**; the solution is instead obtained with:

$$\mathbf{x}=\mathbf{R}_1\ldots\mathbf{R_n}\mathbf{y}=\mathbf{R}_1\ldots\mathbf{R_n}\mathbf{\Lambda}^{-1}\mathbf{d} \tag{3}$$

Numerically, this series of transformations appears to be time consuming because of the numerous matrix multiplications. However, as is the case with the Thomas Algorithm, this process can be adapted into a faster algorithm, depending on the type of problem. The first step of the column inversion, Equation (2), can be replaced by a sequential term-by-term vector multiplication to perform a forward elimination. As for the transformation applied to the solution vector, Equation (3), the nature of the elementary matrix allows a major simplification:

$$\mathbf{x}=\mathbf{R}_1\ldots\mathbf{R_n}\mathbf{y}=\mathbf{y}-y_n\mathbf{f} \tag{4}$$

where vector **f** is composed of the non-diagonal elements f_i of the column-multiplying elementary matrix $\mathbf{R_i}$. Hence, the n matrix multiplications are simply replaced by a vector subtraction and a scalar multiplication.

3. EXAMPLE OF APPLICATION

3.1. Dual contracting grid

In order to study the various effects of pyrolysis gas within a solid medium, a one-dimensional material response implicit solver with solid ablation and pyrolysis has been developed [5, 6]. Because the solver allows cylindrical and spherical coordinates, the geometry is defined relative to a radius. For a given computing domain, the ablating surface can either be at the inside wall (as in circuit-breaker ablation [7]) or at the outside wall (as a in re-entry vehicle [6]). In order to solve both types of problem, possibly at the same time, the possibility for the ablation occurring at either end of a given wall has been implemented.

The mixture energy equation that describes the heat inside the material is as follows:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathrm{cv}} \rho E\,\mathrm{d}V + \int_{\mathrm{cs}} \phi\rho_g h_g v_g\,\mathrm{d}A + \int_{\mathrm{cs}} \dot{q}''\,\mathrm{d}A - \int_{\mathrm{cs}} \rho h v_{\mathrm{cs}}\,\mathrm{d}A = 0$$

where ρ is the density, E the energy, V the volume, ϕ the porosity of the solid, h the enthalpy, v the velocity, A the area and $\dot{q}''$ the heat flux according to Fourier's Law. Subscripts cv, cs and g are the control volume, the control surface and the gas phase, respectively. The equation is discretized using a control volume finite element method [8] and is solved using an Euler implicit time integrator.

In this equation, the unknowns are the temperature T in each cell, as well as the ablation velocities at both ends, $\dot{s}_0 = v_{\mathrm{cs}}(r=0)$ and $\dot{s}_F = v_{\mathrm{cs}}(r=R)$. The numerical algorithm uses neighboring values to evaluate the temperature. Because the code uses a contracting grid scheme, the recession rate has repercussions on each cell. Therefore, the Jacobian Matrix of the numerical scheme is an opposite-bordered tridiagonal matrix (see Figure (1)). Assuming a grid of 6 points (i.e. 5 control volumes), the system of linear equations is of the form:

$$\begin{bmatrix} e_0 & c_0 & 0 & 0 & 0 & 0 & 0 \\ e_1 & b_1 & c_1 & 0 & 0 & 0 & g_1 \\ e_2 & a_2 & b_2 & c_2 & 0 & 0 & g_2 \\ e_3 & 0 & a_3 & b_3 & c_3 & 0 & g_3 \\ e_4 & 0 & 0 & a_4 & b_4 & c_4 & g_4 \\ e_5 & 0 & 0 & 0 & a_5 & b_5 & g_5 \\ 0 & 0 & 0 & 0 & 0 & a_F & g_F \end{bmatrix} \begin{bmatrix} \dot{s}_0 \\ T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ \dot{s}_F \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_F \end{bmatrix}$$

First, a forward elimination process is performed by applying elementary row operations, more specifically by adding a multiple of one row to another on the augmented matrix. As with the Thomas Algorithm, this process is done by a series of term-by-term vector multiplications (see the Appendix for the details). The linear system of equations thus becomes

$$\begin{bmatrix} e_0 & 0 & 0 & 0 & 0 & 0 & g_0 \\ e_1 & b_1 & 0 & 0 & 0 & 0 & g_1 \\ e_2 & a_2 & b_2 & 0 & 0 & 0 & g_2 \\ e_3 & 0 & a_3 & b_3 & 0 & 0 & g_3 \\ e_4 & 0 & 0 & a_4 & b_4 & 0 & g_4 \\ e_5 & 0 & 0 & 0 & a_5 & b_5 & g_5 \\ 0 & 0 & 0 & 0 & 0 & a_F & g_F \end{bmatrix} \begin{bmatrix} \dot{s}_0 \\ T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ \dot{s}_F \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_F \end{bmatrix}$$

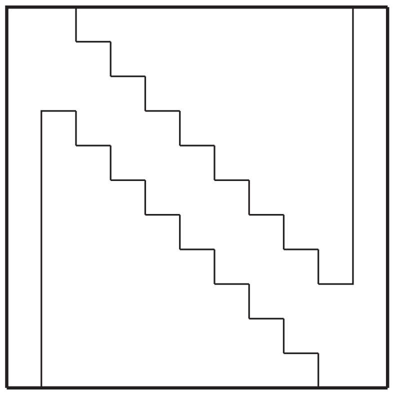
Figure 1. Opposite-bordered tridiagonal matrix.

Although the system is now easier to solve, it is still not straightforward since there are no rows containing only one non-zero elements (i.e. the matrix is not triangular). In order to make sure that all but the corner element of the 7th column disappear, elementary column operations need to be performed on the matrix. The first step is to remove element g_0 using the first column; to do so, the elementary matrix $\mathbf{R}_0$ is applied to the right-hand side of the matrix, and $\mathbf{R}_0^{-1}$ to the solution vector. Explicitly, this elementary column operation matrix is as follows:

$$\mathbf{R}_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -f_0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where $f_0 = g_0/e_0$. The same technique is used to removed g_1 using the second column, and so forth. After all those operations, the linear system of equations becomes

$$\begin{bmatrix} e_0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_1 & b_1 & 0 & 0 & 0 & 0 & 0 \\ e_2 & a_2 & b_2 & 0 & 0 & 0 & 0 \\ e_3 & 0 & a_3 & b_3 & 0 & 0 & 0 \\ e_4 & 0 & 0 & a_4 & b_4 & 0 & 0 \\ e_5 & 0 & 0 & 0 & a_5 & b_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_F & g_F \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_F \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_F \end{bmatrix}$$

It is to be noted that even though these column operations ultimately change only one element of the matrix (element g_F), they still need to be performed because the inverse of those operations need to be re-applied to the solution vector, as shown in Equation (2). In the algorithm, each one of these operations needs to be kept in memory, in the order they were performed. Since the matrix is now lower triangular, the system can be solved directly for $\mathbf{y}$ using a slightly modified version of Thomas' backward substitution, used to solve singly bordered tridiagonal matrices [9].

For the last step, the column operations that were performed on the matrix need to be re-applied inversely to vector $\mathbf{y}$. As explained earlier, this process can be simplified by multiplying all the elementary column matrices and applying this summation matrix to vector **y**:

$$\mathbf{R}_0\mathbf{R}_1\mathbf{R}_2\mathbf{R}_3\mathbf{R}_4\mathbf{R}_5\mathbf{y}=\begin{bmatrix}1&0&0&0&0&0&-f_0\\0&1&0&0&0&0&-f_1\\0&0&1&0&0&0&-f_2\\0&0&0&1&0&0&-f_3\\0&0&0&0&1&0&-f_4\\0&0&0&0&0&1&-f_5\\0&0&0&0&0&0&1\end{bmatrix}\begin{bmatrix}y_0\\y_1\\y_2\\y_3\\y_4\\y_5\\y_F\end{bmatrix}$$

Finally, as seen in Equation (4), the solution of the linear systems of equations breaks down as follows:

$$\begin{bmatrix}\dot{s}_0\\T_1\\T_2\\T_3\\T_4\\T_5\\\dot{s}_F\end{bmatrix}=\begin{bmatrix}y_0\\y_1\\y_2\\y_3\\y_4\\y_5\\y_F\end{bmatrix}-y_F\begin{bmatrix}f_0\\f_1\\f_2\\f_3\\f_4\\f_5\\0\end{bmatrix}$$

Because of the simple nature of the column-multiplication matrices $\mathbf{R_i}$, the proposed method takes only $\mathcal{O}(n)$ operations, as with the regular Thomas Algorithm.

3.2. Other matrix configurations

The proposed algorithm may be applied to solve many types of linear systems of equations, more specifically, when the non-tridiagonal elements of the matrix are organized into columns; some examples are shown in Figure 2. The method can also be adapted to solve block tridiagonal systems, using the methodology presented in [3], for instance.

4. COMPARISON

The Sherman–Morrison–Woodbury algorithm [4] can be used to solve near-tridiagonal systems of linear equations. This algorithm is based on the idea that a small perturbation in a linear system of equations should not change too much the difficulty to solve it. More precisely, if $\mathbf{Ax}=\mathbf{b}$ is easy to solve (or already known), the solution of $\mathbf{Bx}=\mathbf{d}$, where $\mathbf{B}=\mathbf{A}+\mathbf{uv}^{\mathrm{T}}$, can be calculated using

$$\mathbf{x}=\mathbf{y}-\left[\frac{\mathbf{v}\cdot\mathbf{y}}{1+\mathbf{v}\cdot\mathbf{z}}\right]\mathbf{z} \tag{5}$$

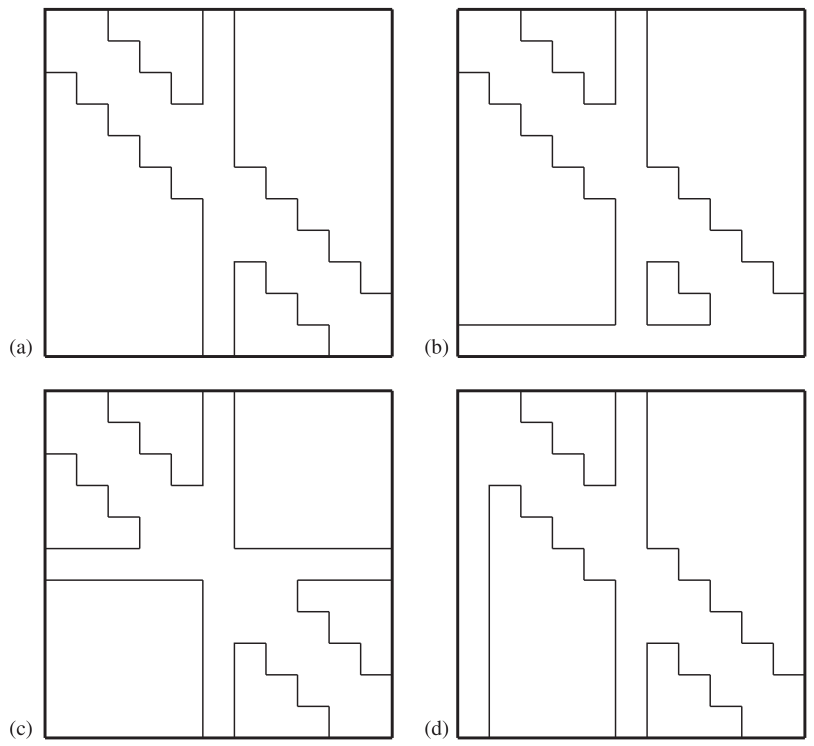

Figure 2. Different configurations of tridiagonal linear systems of linear equations that could be solved with the proposed algorithm: (a) split tridiagonal matrix; (b) split-bordered tridiagonal matrix; (c) cross tridiagonal matrix; and (d) split-bordered tridiagonal matrix.

where $\mathbf{Ay}=\mathbf{d}$ and $\mathbf{Az}=\mathbf{u}$. In the example presented in Section 3.1, perturbation vectors $\mathbf{u}$ and $\mathbf{v}$ could be defined as

$$\mathbf{u}=\begin{bmatrix} 0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ \gamma \end{bmatrix}, \quad \mathbf{v}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

where γ is arbitrary. Matrix $\mathbf{A}$ is then easily inverted to find $\mathbf{y}$ and $\mathbf{z}$ using a modified Thomas Algorithm for a singly bordered tridiagonal system of equations, as the one presented in [9]. Because of the simplicity of vector $\mathbf{v}$, the dot products of Equation (5) are straightforward, and the solution essentially required to solve two singly bordered tridiagonal systems of equations. Therefore, it is clear to see that, as opposed to the proposed method, the Sherman–Morrison–Woodbury algorithm has two drawbacks; the difficulty of finding the perturbation vectors and the time required to solve twice the reduced matrix.

Table I shows the number of floating point operations for a linear system of n equations for the proposed algorithm, the Sherman–Morrison–Woodbury algorithm and other well-known algorithms. As can be seen, on top of avoiding the aforementioned drawbacks, the proposed algorithm shows a notable increase in resolution speed (almost 20%). It is obvious that the values presented here are not absolute since they depend on the exact algorithm used; however, the general tendency should remain the same.

Table I. Number of floating point operations for a linear system of n equations for different algorithms.

Algorithm type	Reference	Number of floating point operations
Tridiagonal Thomas	Reference [1]	$8n-7$
Cyclic Thomas	Reference [10]	$17n-16$
Singly bordered Thomas	Reference [9]	$12n-19$
Opposite-bordered Thomas	Current work	$21n-39$
Sherman–Morrison–Woodbury	Reference [4]	$25n-14$

5. CONCLUSION

A new and efficient algorithm for the resolution of a certain type of near-tridiagonal linear system of equations has been proposed. This new algorithm may be applied to solve many types of linear systems of equations, more specifically when the non-tridiagonal elements of the matrix are organized into columns. The algorithm takes advantage of the properties of elementary column operation matrices to allow for a very efficient resolution. A specific applied example is shown, using the numerical solution of an ablation and heat transfer problem. Finally, a comparison with an already existing algorithm, the Sherman–Morrison–Woodbury Algorithm, is presented; because of the properties of elementary column operations, the proposed algorithm demonstrates a reduced number of floating point operations. On top of showing excellent results, the algorithm opens up a new way to solve efficiently near-tridiagonal systems of equations.

APPENDIX

For the problem discussed in Section 3.1, the FORTRAN pseudo-code is:

```
c    !FORWARD ELIMINATION
     DO j=N-1,1,-1
        f = -c(j)/b(j+1)
        b(j) =  b(j) + f*a(j+1)
        e(j) =  e(j) + f*e(j+1)
        g(j) =  g(j) + f*g(j+1)
        d(j) =  d(j) + f*d(j+1)
     END DO
     f = -c0/b(1)
     e0 = e0 + f*e(1)
     d0 = d0 + f*d(1)
     g0 =  f*g(1)

c    !END COLUMN ELIMINATION
     fc0 = -g0/e0
     DO j = 1,N
        g(j) = g(j) + fc0*e(j)
     ENDDO
     DO j = 1,N-1
        fc(j) = -g(j)/b(j)
        g(j+1) = g(j+1) + fc(j)*a(j+1)
     ENDDO
     fc(N) = -g(N)/b(N)
     gF = gF + fc(N)*aF

c    !BACK SUBSTITUTION
     S0 = d0/e0
     T(1) = (d(1) - e(1)*S0)/b(1)
     DO j=2,N
        T(j) = ( d(j) - e(j)*S0 - a(j)*T(j-1) ) / b(j)
     END DO
     SF = ( dF - aF*T(N) ) / gF

c    !COLUMN CORRECTION
     S0 = S0 + SF*fc0
     DO j=1,N
        T(j) =  T(j) + SF*fc(j)
     END DO
```

ACKNOWLEDGEMENTS

The authors would like to thank the *Government of Québec* which, through the *Fonds de recherche sur la nature et les technologies* (*NATEQ*), provides a fellowship to the first author. Additional funding is provided by the *Constellation University Institutes Program*, under NASA grant NCC3-989.

REFERENCES

1. Thomas LH. Elliptic problems in linear difference equations over a network. *Watson Science Computer Laboratory Report*, Columbia University, NY, 1949.
2. Bieniasz L. Extension of the Thomas algorithm to a class of algebraic linear equation systems involving quasi-block-tridiagonal matrices with isolated block-pentadiagonal rows, assuming variable block dimension. *Computing* 2001; **37**:269–285.
3. Batista. A cyclic block-tridiagonal solver. *Advances in Engineering Software* 2006; **37**:69–74.
4. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. *Numerical Recipes*: *The Art of Scientific Computing* (2nd edn). Cambridge University Press: Cambridge, 1992.
5. Amar AJ, Blackwell BF, Edward JR. One-dimensional ablation with pyrolysis gas flow using a full Newton's method and finite control volume procedure. *39th AIAA Thermophysics Conference*, AIAA-2007-4535, Miami, FL, 2007; 41.
6. Martin A, Boyd ID. Simulation of pyrolysis gas within a thermal protection system. *40th AIAA Thermophysics Conference*. AIAA: Seattle, WA, 2008.
7. Martin A, Reggio M, Trepanier JY, Guo X. Transient ablation regime in circuit breakers. *Plasma Science and Technology* 2007; **9**(6):653–656.
8. Amar AJ, Blackwell BF, Edward JR. One-dimensional ablation using a full Newton's method and finite control volume procedure. *9th AIAA/ASME Joint Thermophysics and Heat Transfer Conference*, AIAA-2006-2910, San Francisco, CA, 2006; 26.
9. Amar AJ. Modeling of one-dimensional ablation with porous flow using finite control volume procedure. *Master's Thesis*, North Carolina State University, Raleigh, 2006.
10. Sebben S, Baliga B. Some extensions of tridiagonal and pentadiagonal matrix algorithms. *Numerical Heat Transfer*, *Part B*: *Fundamentals* 1995; **28**(4):323–351.