

Unconditionally Stable Finite Difference Scheme and Iterative Solution of 2D Microscale Heat Transport Equation *

Jun Zhang[†]

Laboratory for High Performance Scientific Computing and Computer Simulation
Department of Computer Science
University of Kentucky
773 Anderson Hall
Lexington, KY 40506-0046, USA

Jennifer J. Zhao[‡]

Department of Mathematics and Statistics
University of Michigan at Dearborn
Dearborn, MI 48128-1491, USA

February 21, 2001

Abstract

A two dimensional time dependent heat transport equation at the microscale is derived. A second order finite difference scheme in both time and space is introduced and the unconditional stability of the finite difference scheme is proved. A computational procedure is designed to solve the discretized linear system at each time step by using a preconditioned Conjugate Gradient method. Numerical results are presented to validate the accuracy of the finite difference scheme and the efficiency of the proposed computational procedure.

Key words: Heat transport equation, finite difference scheme, preconditioned Conjugate Gradient, Crank-Nicholson integrator.

Mathematical Subject Classification: 65M06, 65N12.

1 Introduction

Microtechnologies based on high rate heating on thin film structures are developing rapidly in recent years due to the advancement of short pulse laser technologies and their appli-

*This paper has been accepted for publication in *Journal of Computational Physics*.

[†]E-mail: jzhang@cs.uky.edu. URL: <http://www.cs.uky.edu/~jzhang>. The research of this author was supported in part by the U.S. National Science Foundation under grants CCR-9902022, CCR-9988165 and CCR-0043861, and in part by the University of Kentucky Center for Computational Sciences.

[‡]E-mail: xich@umd.umich.edu. URL: <http://www.umd.umich.edu/~xich>.

cations to micromanufacturing processes [1, 3, 13]. These microtechnology applications frequently deal with thermal behavior of thin films [10]. Components of microelectronic devices such as thin films of metals, dielectrics, e.g., SiO₂ or Si semiconductors, can be simulated using computers instead of actual prototyping. The demand on fast switching speed of electronic devices has pushed for the reduction of the device size to microscale. The side effect of device size reduction is the increase of heat generation rate that leads to a higher thermal load on the microdevice. Studying the thermal behavior of thin films is essential for predicting the performance of a microelectronic device or for designing the desired microstructure.

The microscale heat transport equation arises from many applications, e.g., from phonon electron interaction model [17], the single energy equation [21, 22], the phonon scattering model [9], the phonon radiative transfer model [10], and the lagging behavior model [14, 21, 20].

Numerical solution of one dimensional microscale heat transport equation has been considered by a few authors. Qiu and Tien [16] used the Crank-Nicholson finite difference scheme for solving the phonon electron interaction model. Joshi and Majumdar [10] solved the phonon radiative transfer model in an one dimensional medium by using the explicit upstream differencing method. Özisik and Tzou [14] studied the lagging behavior by solving the equation in a semi-infinite interval. Dai and Nassar [5] considered the numerical solution of the microscale heat transport equation in a finite interval $x \in [0, \epsilon]$, where the unit of ϵ is in microscale. Recently, we employed a fourth order compact finite difference discretization scheme to solve the one dimensional microscale heat transport equation and obtained highly accurate numerical solution [23].

In this paper, we generalize the microscale heat transport equation to two dimensions and propose a set of numerical strategies to solve the governing equation efficiently. The heat transport equations used to describe the thermal behavior of microstructure can be expressed as:

$$\begin{aligned} -\nabla \cdot \vec{q} + Q &= \rho C_p \frac{\partial T}{\partial t}, \\ \vec{q}(x, y, t + \tau_q) &= -k \nabla T(x, y, t + \tau_T), \end{aligned} \quad (1)$$

where $\vec{q} = (q_1, q_2)$ is the heat flux, and (q_1, q_2) are the heat flux components in the x and y directions, respectively. T is the temperature, k is conductivity, ρ is density, C_p is the specific heat, and Q is a heat source. τ_q and τ_T are positive constants which are the time lags of the heat flux and temperature gradient, respectively. In the classical theory of diffusion, the heat flux vector (\vec{q}) and the temperature gradient (∇T) across a material volume are assumed to occur at the same instant of time. They satisfy the Fourier's law of heat conduction:

$$\vec{q}(x, y, t) = -k \nabla T(x, y, t). \quad (2)$$

However, if the scale in one direction is at microscale, i.e., is of order $0.1\mu\text{m}$, then the heat flux and temperature gradient in this direction will occur at different times. The heat flux and the temperature gradient in the microscale direction satisfy (1) instead of

(2). This is the so called lagging effect [21]. Using Taylor series expansions, the first order approximation of (1) can be written as

$$\vec{q} + \tau_q \frac{\partial \vec{q}}{\partial t} = -k \left[\nabla T + \tau_T \frac{\partial}{\partial t} [\nabla T] \right]. \quad (3)$$

If we consider a film with width to be an order of $0.1\mu\text{m}$ and the length to be an order of 1mm , then the component of the heat flux in the x direction satisfies the traditional Fourier's law, while the component in the y direction satisfies (1). Hence, we have

$$q_1 = -k \frac{\partial T}{\partial x}, \quad (4)$$

$$q_2 + \tau_q \frac{\partial q_2}{\partial t} = -k \left[\frac{\partial T}{\partial y} + \tau_T \frac{\partial}{\partial t} \left(\frac{\partial T}{\partial y} \right) \right]. \quad (5)$$

In two dimensions, Equation (1) can be written as

$$-\frac{\partial q_2}{\partial y} = \rho C_p \frac{\partial T}{\partial t} + \frac{\partial q_1}{\partial x} - Q.$$

Substituting (4) into the above equation and simplify, we obtain

$$\frac{\partial q_2}{\partial y} = -\rho C_p \frac{\partial T}{\partial t} + k \frac{\partial^2 T}{\partial x^2} + Q. \quad (6)$$

Differentiating (5) with respect to y yields

$$\frac{\partial q_2}{\partial y} + \tau_q \frac{\partial^2 q_2}{\partial t \partial y} = -k \left[\frac{\partial^2 T}{\partial y^2} + \tau_T \frac{\partial}{\partial t} \left(\frac{\partial^2 T}{\partial y^2} \right) \right]. \quad (7)$$

Substituting (6) into (7) and after simplification, we obtain the following equation:

$$\begin{aligned} & \rho C_p \left(\frac{\partial T}{\partial t} + \tau_q \frac{\partial^2 T}{\partial t^2} \right) \\ &= k \frac{\partial}{\partial t} \left(\tau_q \frac{\partial^2 T}{\partial x^2} + \tau_T \frac{\partial^2 T}{\partial y^2} \right) + k \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) + \left(Q + \tau_q \frac{\partial Q}{\partial t} \right). \end{aligned}$$

Denote $\alpha = k/\rho C_p$ and $S = (Q + \tau_q \frac{\partial Q}{\partial t})/k$, we have the governing two dimensional microscale heat transport equation in the form of

$$\frac{1}{\alpha} \frac{\partial T}{\partial t} + \frac{\tau_q}{\alpha} \frac{\partial^2 T}{\partial t^2} = \tau_q \frac{\partial^3 T}{\partial t \partial x^2} + \tau_T \frac{\partial^3 T}{\partial t \partial y^2} + \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + S. \quad (8)$$

The initial and boundary conditions are:

$$\begin{aligned} T(x, y, 0) &= T_0(x, y), & \frac{\partial T}{\partial t}(x, y, 0) &= T_1(x, y), \\ T(0, y, t) &= T_2(y, t), & T(L, y, t) &= T_3(y, t), \\ T(x, 0, t) &= T_4(x, t), & T(x, \epsilon, t) &= T_5(x, t). \end{aligned}$$

The parameters α, τ_q and τ_T in Eq. (8) have their physical domains of definition related to certain material properties in microscale heat transfer. They are left as free parameters in our study for more general testing of elliptic, hyperbolic and parabolic cases, and for numerical testing of our proposed discretization scheme and preconditioned iterative solution method.

The rest of the paper describes techniques that we propose to solve the microscale heat transport equation (8). In particular, we introduce a finite difference scheme to discretize (8) in Section 2. In Section 3, we prove the unconditional stability of the finite difference scheme with respect to initial values. A preconditioned Conjugate Gradient iterative method is discussed in Section 4 to solve the sparse linear systems arising at each time step. In Section 5 experimental results are presented to validate the proposed numerical techniques. Concluding remarks are given in Section 6.

2 Finite Difference Discretization

It is commented by Dai and Nassar [5] that direct discretization of (8) leads to a finite difference scheme that is three level in time. In such a case, stability of the discretization scheme may be difficult to ascertain. To avoid a three level discretization scheme, following the idea of Dai and Nassar, we introduce an auxiliary function as [5]

$$\theta = T + \tau_q \frac{\partial T}{\partial t}, \quad (9)$$

and change (8) into

$$\frac{1}{\alpha} \frac{\partial \theta}{\partial t} = \frac{\partial}{\partial t} \left(\tau_q \frac{\partial^2 T}{\partial x^2} + \tau_T \frac{\partial^2 T}{\partial y^2} \right) + \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + S. \quad (10)$$

The initial and boundary conditions can be reformulated as:

$$\begin{aligned} T(x, y, 0) &= T_0, & T(0, y, t) &= T_2, & T(L, y, t) &= T_3, & T(x, 0, t) &= T_4, \\ T(x, \epsilon, t) &= T_5, & \theta(x, y, 0) &= T_0 + \tau_q T_1, & \theta(0, y, t) &= T_2 + \tau_q \frac{\partial T_2}{\partial t}, \\ \theta(L, y, t) &= T_3 + \tau_q \frac{\partial T_3}{\partial t}, & \theta(x, 0, t) &= T_4 + \tau_q \frac{\partial T_4}{\partial t}, & \theta(x, \epsilon, t) &= T_5 + \tau_q \frac{\partial T_5}{\partial t}. \end{aligned}$$

Now consider a spatial domain $\Omega = [0, L] \times [0, \epsilon]$, where ϵ is at the microscale of order $0.1\mu\text{m}$, and $\epsilon \ll L$. Let Ω be discretized with uniform meshes with grid points at $x_i = i\Delta x, y_j = j\Delta y, i, j = 0, 1, \dots, N$, where $N\Delta x = L$ and $N\Delta y = \epsilon$. The number of grid points in the x and y directions could be different. For convenience we only consider the case that they are equal.

Denote the standard central difference operators as

$$\delta_x^2 T_{i,j} = \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{\Delta x^2} \quad \text{and} \quad \delta_y^2 T_{i,j} = \frac{T_{i,j-1} - 2T_{i,j} + T_{i,j+1}}{\Delta y^2}, \quad (11)$$

which are of accuracy order $O(\Delta x^2)$ and $O(\Delta y^2)$, respectively. Note that $\Delta x \gg \Delta y$ in general.

We discretize (10) using a Crank-Nicholson type integrator with the central difference operators (11) at the time step $(n + \frac{1}{2})\Delta t$ as

$$\begin{aligned} \frac{1}{\alpha\Delta t}(\theta_{i,j}^{n+1} - \theta_{i,j}^n) &= \frac{1}{\Delta t} \left[(\tau_q \delta_x^2 T_{i,j}^{n+1} + \tau_T \delta_y^2 T_{i,j}^{n+1}) - (\tau_q \delta_x^2 T_{i,j}^n + \tau_T \delta_y^2 T_{i,j}^n) \right] \\ &+ \frac{1}{2} \left[(\delta_x^2 T_{i,j}^{n+1} + \delta_y^2 T_{i,j}^{n+1}) + (\delta_x^2 T_{i,j}^n + \delta_y^2 T_{i,j}^n) \right] + S_{i,j}^{n+\frac{1}{2}}, \end{aligned} \quad (12)$$

where Δt is the uniform time step size. This scheme obviously has an accuracy order of $O(\Delta t^2 + \Delta x^2 + \Delta y^2)$.

Equation (9) can be discretized by a second order trapezoidal method with respect to time t

$$\frac{1}{2}(\theta_{i,j}^{n+1} + \theta_{i,j}^n) = \frac{1}{2}(T_{i,j}^{n+1} + T_{i,j}^n) + \frac{\tau_q}{\Delta t}(T_{i,j}^{n+1} - T_{i,j}^n), \quad (13)$$

which can be solved for $\theta_{i,j}^{n+1}$ as

$$\theta_{i,j}^{n+1} = \left(1 + \frac{2\tau_q}{\Delta t}\right) T_{i,j}^{n+1} + \left(1 - \frac{2\tau_q}{\Delta t}\right) T_{i,j}^n - \theta_{i,j}^n. \quad (14)$$

Substituting (14) into (12) and after simplification, we have

$$\begin{aligned} &\left(\frac{\tau_q}{\Delta t} + \frac{1}{2}\right) \delta_x^2 T_{i,j}^{n+1} + \left(\frac{\tau_T}{\Delta t} + \frac{1}{2}\right) \delta_y^2 T_{i,j}^{n+1} - \frac{1}{\alpha\Delta t} \left(1 + \frac{2\tau_q}{\Delta t}\right) T_{i,j}^{n+1} \\ &= \frac{1}{\alpha\Delta t} \left[\left(1 - \frac{2\tau_q}{\Delta t}\right) T_{i,j}^n - 2\theta_{i,j}^n \right] + \left(\frac{\tau_q}{\Delta t} - \frac{1}{2}\right) \delta_x^2 T_{i,j}^n + \left(\frac{\tau_T}{\Delta t} - \frac{1}{2}\right) \delta_y^2 T_{i,j}^n - S_{i,j}^{n+\frac{1}{2}}, \end{aligned} \quad (15)$$

which will be used to compute $T_{i,j}^{n+1}$. Then the computed $T_{i,j}^{n+1}$ is substituted into (14) to compute $\theta_{i,j}^{n+1}$. The corresponding discretized initial and boundary conditions are given as:

$$T_{i,j}^0 = (T_0)_{i,j}, \quad T_{0,j}^n = (T_2)_j^n, \quad T_{N,j}^n = (T_3)_j^n, \quad T_{i,0}^n = (T_4)_i^n, \quad T_{i,N}^n = (T_5)_i^n, \quad (16)$$

$$\theta_{i,j}^0 = (T_0 + \tau_q T_1)_{i,j}, \quad \theta_{0,j}^n = \left(T_2 + \tau_q \frac{\partial T_2}{\partial t}\right)_j^n, \quad \theta_{N,j}^n = \left(T_3 + \tau_q \frac{\partial T_3}{\partial t}\right)_j^n, \quad (17)$$

$$\theta_{i,0}^n = \left(T_4 + \tau_q \frac{\partial T_4}{\partial t}\right)_i^n, \quad \theta_{i,N}^n = \left(T_5 + \tau_q \frac{\partial T_5}{\partial t}\right)_i^n. \quad (18)$$

3 Stability Analysis

We shall prove the unconditional stability of the finite difference schemes (12) and (13) with respect to the initial values. The technique that we will use in our proof is the discrete energy method [4, 11]. To this end, we denote by G the set of discrete values

$$\left\{ u^n = \left\{ u_{i,j}^n \right\}, \text{ with } u_{0,j}^n = u_{N,j}^n = u_{i,0}^n = u_{i,N}^n = 0. \quad 1 \leq i, j \leq N. \right\}.$$

We then make the following norm definitions for any $u^n, v^n \in G$,

$$(u^n, v^n) = \Delta x^2 \sum_{i,j=1}^{N-1} u_{i,j}^n v_{i,j}^n, \quad \|u^n\|^2 = (u^n, u^n).$$

The following results can be verified easily [4, 11].

Lemma 3.1 *For any $u^n, v^n \in G$, the following equalities hold*

$$(\delta_x^2 u^n, v^n) = -(\delta_x u^n, \delta_x v^n), \quad (\delta_y^2 u^n, v^n) = -(\delta_y u^n, \delta_y v^n),$$

where

$$\delta_x u_{i,j}^n = \frac{u_{i+1,j}^n - u_{i,j}^n}{\Delta x}, \quad \delta_y u_{i,j}^n = \frac{u_{i,j+1}^n - u_{i,j}^n}{\Delta y},$$

are the forward difference operators.

Theorem 3.2 *Suppose that $\{T_{i,j}^n, \theta_{i,j}^n\}$ and $\{V_{i,j}^n, \xi_{i,j}^n\}$ are the solutions of the finite difference schemes (12) and (13) which satisfy the boundary conditions (16) - (18), and have different initial values $\{T_{i,j}^0, \theta_{i,j}^0\}$ and $\{V_{i,j}^0, \xi_{i,j}^0\}$, respectively. Let $w_{i,j}^n = \theta_{i,j}^n - \xi_{i,j}^n$, $\epsilon_{i,j}^n = T_{i,j}^n - V_{i,j}^n$, then $\{w_{i,j}^n, \epsilon_{i,j}^n\}$ satisfy*

$$\begin{aligned} & \frac{1}{\alpha} \|w^n\|^2 + 2\tau_q \|\delta_x \epsilon^n\|^2 + (\tau_q + \tau_T) \|\delta_y \epsilon^n\|^2 \\ & \leq \frac{1}{\alpha} \|w^0\|^2 + 2\tau_q \|\delta_x \epsilon^0\|^2 + (\tau_q + \tau_T) \|\delta_y \epsilon^0\|^2, \end{aligned} \quad (19)$$

for any $0 \leq n\Delta t \leq t_{\text{stop}}$. This implies that the finite difference scheme is unconditionally stable with respect to the initial values.

Proof: Since $\{T_{i,j}^n, \theta_{i,j}^n\}$ and $\{V_{i,j}^n, \xi_{i,j}^n\}$ are both the solutions of (12) with the same boundary conditions, so $\{w^n, \epsilon^n\} \in G$, and they also satisfy

$$\begin{aligned} \frac{1}{\alpha \Delta t} (w_{i,j}^{n+1} - w_{i,j}^n) &= \frac{1}{\Delta t} (\tau_q \delta_x^2 \epsilon_{i,j}^{n+1} + \tau_T \delta_y^2 \epsilon_{i,j}^{n+1} - \tau_q \delta_x^2 \epsilon_{i,j}^n - \tau_T \delta_y^2 \epsilon_{i,j}^n) \\ &+ \frac{1}{2} (\delta_x^2 \epsilon_{i,j}^{n+1} + \delta_y^2 \epsilon_{i,j}^{n+1} + \delta_x^2 \epsilon_{i,j}^n + \delta_y^2 \epsilon_{i,j}^n) \\ &= \frac{1}{2} \delta_x^2 (\epsilon_{i,j}^{n+1} + \epsilon_{i,j}^n) + \frac{1}{2} \delta_y^2 (\epsilon_{i,j}^{n+1} + \epsilon_{i,j}^n) \\ &+ \frac{\tau_q}{\Delta t} \delta_x^2 (\epsilon_{i,j}^{n+1} - \epsilon_{i,j}^n) + \frac{\tau_T}{\Delta t} \delta_y^2 (\epsilon_{i,j}^{n+1} - \epsilon_{i,j}^n). \end{aligned} \quad (20)$$

From (13), we have

$$w_{i,j}^{n+1} + w_{i,j}^n = (\epsilon_{i,j}^{n+1} + \epsilon_{i,j}^n) + \frac{2\tau_q}{\Delta t} (\epsilon_{i,j}^{n+1} - \epsilon_{i,j}^n). \quad (21)$$

We can easily get the following results by using (21) and the definitions.

$$(\epsilon^{n+1} + \epsilon^n, w^{n+1} + w^n) = \|\epsilon^{n+1} + \epsilon^n\|^2 + \frac{2\tau_q}{\Delta t} (\|\epsilon^{n+1}\|^2 - \|\epsilon^n\|^2), \quad (22)$$

$$(\epsilon^{n+1} - \epsilon^n, w^{n+1} + w^n) = \frac{2\tau_q}{\Delta t} (\|\epsilon^{n+1} - \epsilon^n\|^2) + \|\epsilon^{n+1}\|^2 - \|\epsilon^n\|^2, \quad (23)$$

and

$$\begin{aligned} & (\delta_x(\epsilon^{n+1} + \epsilon^n), \delta_x(w^{n+1} + w^n)) \\ &= \|\delta_x(\epsilon^{n+1} + \epsilon^n)\|^2 + \frac{2\tau_q}{\Delta t} \left(\|\delta_x \epsilon^{n+1}\|^2 - \|\delta_x \epsilon^n\|^2 \right), \end{aligned} \quad (24)$$

$$\begin{aligned} & (\delta_x(\epsilon^{n+1} - \epsilon^n), \delta_x(w^{n+1} + w^n)) \\ &= \frac{2\tau_q}{\Delta t} \|\delta_x(\epsilon^{n+1} - \epsilon^n)\|^2 + \|\delta_x \epsilon^{n+1}\|^2 - \|\delta_x \epsilon^n\|^2, \end{aligned} \quad (25)$$

$$\begin{aligned} & (\delta_y(\epsilon^{n+1} + \epsilon^n), \delta_y(w^{n+1} + w^n)) \\ &= \|\delta_y(\epsilon^{n+1} + \epsilon^n)\|^2 + \frac{2\tau_q}{\Delta t} \left(\|\delta_y \epsilon^{n+1}\|^2 - \|\delta_y \epsilon^n\|^2 \right), \end{aligned} \quad (26)$$

$$\begin{aligned} & (\delta_y(\epsilon^{n+1} - \epsilon^n), \delta_y(w^{n+1} + w^n)) \\ &= \frac{2\tau_q}{\Delta t} \|\delta_y(\epsilon^{n+1} - \epsilon^n)\|^2 + \|\delta_y \epsilon^{n+1}\|^2 - \|\delta_y \epsilon^n\|^2, \end{aligned} \quad (27)$$

We now multiply both sides of (20) by $(w_{i,j}^{n+1} + w_{i,j}^n) \Delta x^2$ and sum over i and j to get

$$\begin{aligned} & \frac{1}{\alpha \Delta t} \left(\|w^{n+1}\|^2 - \|w^n\|^2 \right) \\ &= \frac{1}{2} \left(\delta_x^2(\epsilon^{n+1} + \epsilon^n), w^{n+1} + w^n \right) + \frac{1}{2} \left(\delta_y^2(\epsilon^{n+1} + \epsilon^n), w^{n+1} + w^n \right) \\ & \quad + \frac{\tau_q}{\Delta t} \left(\delta_x^2(\epsilon^{n+1} - \epsilon^n), w^{n+1} + w^n \right) + \frac{\tau_q}{\Delta t} \left(\delta_y^2(\epsilon^{n+1} - \epsilon^n), w^{n+1} + w^n \right). \end{aligned} \quad (28)$$

We estimate each term on the right-hand side of (28). Using Lemma 3.1 and (24), we can get

$$\begin{aligned} & \frac{1}{2} \left(\delta_x^2(\epsilon^{n+1} + \epsilon^n), w^{n+1} + w^n \right) \\ &= -\frac{1}{2} \left(\delta_x(\epsilon^{n+1} + \epsilon^n), \delta_x(w^{n+1} + w^n) \right) \\ &= -\frac{1}{2} \left[\|\delta_x(\epsilon^{n+1} + \epsilon^n)\|^2 + \frac{2\tau_q}{\Delta t} \left(\|\delta_x \epsilon^{n+1}\|^2 - \|\delta_x \epsilon^n\|^2 \right) \right]. \end{aligned} \quad (29)$$

Similarly, using Lemma 3.1 and (26), we have

$$\begin{aligned} & \frac{1}{2} \left(\delta_y^2(\epsilon^{n+1} + \epsilon^n), w^{n+1} + w^n \right) \\ &= -\frac{1}{2} \left[\|\delta_y(\epsilon^{n+1} + \epsilon^n)\|^2 + \frac{2\tau_q}{\Delta t} \left(\|\delta_y \epsilon^{n+1}\|^2 - \|\delta_y \epsilon^n\|^2 \right) \right]. \end{aligned} \quad (30)$$

Using (25) and (27), we can get the following results for the other two terms on the right-hand side of (28).

$$\begin{aligned} & \frac{\tau_q}{\Delta t} \left(\delta_x^2(\epsilon^{n+1} - \epsilon^n), w^{n+1} + w^n \right) \\ &= -\frac{\tau_q}{\Delta t} \left(\frac{2\tau_q}{\Delta t} \|\delta_x(\epsilon^{n+1} - \epsilon^n)\|^2 + \|\delta_x \epsilon^{n+1}\|^2 - \|\delta_x \epsilon^n\|^2 \right). \end{aligned} \quad (31)$$

$$\begin{aligned}
& \frac{\tau_T}{\Delta t} \left(\delta_y^2 (\epsilon^{n+1} - \epsilon^n), w^{n+1} + w^n \right) \\
&= -\frac{\tau_T}{\Delta t} \left(\frac{2\tau_q}{\Delta t} \|\delta_y(\epsilon^{n+1} - \epsilon^n)\|^2 + \|\delta_y \epsilon^{n+1}\|^2 - \|\delta_y \epsilon^n\|^2 \right). \tag{32}
\end{aligned}$$

Substituting (29), (30), (31) and (32) back into (28), we have

$$\begin{aligned}
& \frac{1}{\alpha \Delta t} \left(\|w^{n+1}\|^2 - \|w^n\|^2 \right) \\
&= -\frac{1}{2} \left[\|\delta_x(\epsilon^{n+1} + \epsilon^n)\|^2 + \frac{2\tau_q}{\Delta t} \left(\|\delta_x \epsilon^{n+1}\|^2 - \|\delta_x \epsilon^n\|^2 \right) \right] \\
&\quad -\frac{1}{2} \left[\|\delta_y(\epsilon^{n+1} + \epsilon^n)\|^2 + \frac{2\tau_q}{\Delta t} \left(\|\delta_y \epsilon^{n+1}\|^2 - \|\delta_y \epsilon^n\|^2 \right) \right] \\
&\quad -\frac{\tau_q}{\Delta t} \left(\frac{2\tau_q}{\Delta t} \|\delta_x(\epsilon^{n+1} - \epsilon^n)\|^2 + \|\delta_x \epsilon^{n+1}\|^2 - \|\delta_x \epsilon^n\|^2 \right) \\
&\quad -\frac{\tau_T}{\Delta t} \left(\frac{2\tau_q}{\Delta t} \|\delta_y(\epsilon^{n+1} - \epsilon^n)\|^2 + \|\delta_y \epsilon^{n+1}\|^2 - \|\delta_y \epsilon^n\|^2 \right). \tag{33}
\end{aligned}$$

Simply dropping the four negative terms from the right-hand side of (33) yields

$$\begin{aligned}
\frac{1}{\alpha \Delta t} \left(\|w^{n+1}\|^2 - \|w^n\|^2 \right) &\leq -\frac{2\tau_q}{\Delta t} \|\delta_x \epsilon^{n+1}\|^2 - \frac{\tau_q + \tau_T}{\Delta t} \|\delta_y \epsilon^{n+1}\|^2 \\
&\quad + \frac{2\tau_q}{\Delta t} \|\delta_x \epsilon^n\|^2 + \frac{\tau_q + \tau_T}{\Delta t} \|\delta_y \epsilon^n\|^2,
\end{aligned}$$

which is

$$\begin{aligned}
& \frac{1}{\alpha} \|w^{n+1}\|^2 + 2\tau_q \|\delta_x \epsilon^{n+1}\|^2 + (\tau_q + \tau_T) \|\delta_y \epsilon^{n+1}\|^2 \\
&\leq \frac{1}{\alpha} \|w^n\|^2 + 2\tau_q \|\delta_x \epsilon^n\|^2 + (\tau_q + \tau_T) \|\delta_y \epsilon^n\|^2. \tag{34}
\end{aligned}$$

(19) follows from (34) by recursion with respect to n . \square

4 Solution Strategies

In order to compute the solution of (15) a pentadiagonal matrix needs to be solved at each time step. The solution of this linear system dominates the total simulation cost. Direct solution methods are not usually practical for large values of N due to the excessive memory and computational requirements. A common sparse matrix solution strategy in engineering society is to use either a direct band solver or an alternating direction implicit (ADI) solution scheme [15] to compute the solution at each time step. We propose to use a preconditioned iterative method to solve the sparse linear systems.

Let us multiply both sides of (15) with -1 and rewrite it in a simplified (standard) form

$$aT_{i,j}^{n+1} + b(T_{i-1,j}^{n+1} + T_{i+1,j}^{n+1}) + c(T_{i,j-1}^{n+1} + T_{i,j+1}^{n+1}) = F_{i,j}^n, \tag{35}$$

where the coefficients and the right-hand side are

$$\begin{aligned}
b &= -\frac{1}{\Delta x^2} \left(\frac{\tau_q}{\Delta t} + \frac{1}{2} \right), \\
c &= -\frac{1}{\Delta y^2} \left(\frac{\tau_T}{\Delta t} + \frac{1}{2} \right), \\
a &= \frac{2}{\Delta x^2} \left(\frac{\tau_q}{\Delta t} + \frac{1}{2} \right) + \frac{2}{\Delta y^2} \left(\frac{\tau_T}{\Delta t} + \frac{1}{2} \right) + \frac{1}{\alpha \Delta t} \left(1 + \frac{2\tau_q}{\Delta t} \right) \\
&= \frac{1}{\alpha \Delta t} \left(1 + \frac{2\tau_q}{\Delta t} \right) - 2(b + c), \\
F_{i,j}^n &= \left[\frac{2}{\Delta x^2} \left(\frac{\tau_q}{\Delta t} - \frac{1}{2} \right) + \frac{2}{\Delta y^2} \left(\frac{\tau_T}{\Delta t} - \frac{1}{2} \right) - \frac{1}{\alpha \Delta t} \left(1 - \frac{2\tau_q}{\Delta t} \right) \right] T_{i,j}^n \\
&\quad - \frac{1}{\Delta x^2} \left(\frac{\tau_q}{\Delta t} - \frac{1}{2} \right) (T_{i-1,j}^n + T_{i+1,j}^n) - \frac{1}{\Delta y^2} \left(\frac{\tau_T}{\Delta t} - \frac{1}{2} \right) (T_{i,j-1}^n + T_{i,j+1}^n) \\
&\quad + \frac{2}{\alpha \Delta t} \theta_{i,j}^n + S_{i,j}^{n+\frac{1}{2}}. \tag{36}
\end{aligned}$$

Hence, the system of linear equations (35) is symmetric and strictly diagonally dominant. Let us further scale (35) with a and assemble the linear system of equations as

$$AT = F, \tag{37}$$

where A is the coefficient matrix with 5 diagonals (a unit main diagonal). T is the solution vector and F is the right-hand side vector. Since A is a Stieltjes matrix (symmetric positive definite M-matrix), we should expect a fast convergence rate for most iterative methods used to solve (37). A particular important iterative method for solving symmetric positive definite linear systems is the Conjugate Gradient method [8]. To accelerate convergence rate of the Conjugate Gradient (CG) method, a preconditioner M is usually applied to (37) to transform it into a more favorable form

$$M^{-1}AT = M^{-1}F. \tag{38}$$

The key issue in many such computations is to find a good preconditioner M , which should be inexpensive to compute and a solution with it is easy to realize. The preconditioned Conjugate Gradient method is usually abbreviated as PCG method.

As we remarked previously, the linear system (37) is not difficult to solve for most iterative methods. The important issue that interests us is how to solve it most efficiently in a time dependent situation like in the microscale heat transport simulation. Note that (37) needs to be solved at each time step with the same coefficient matrix A and a different right-hand side F . Thus an iterative method converges slowly will not be suitable. A direct method that does not have much fill-in would be much more appealing. A compromise is to use a fast iterative method coupled with a very accurate and robust preconditioner so that (37) can be solved in a few iterations at each time step.

We implemented one of the simplest preconditioners based on the incomplete Cholesky (IC) factorization of A . So M is constructed with a Cholesky factorization of A , but only

the entries corresponding to the nonzero positions of A are computed and stored [12]. Hence, M is as sparse as A and contains 5 diagonals. This simple IC preconditioner works so well for the current problem that we consider the implementation of other powerful preconditioning strategies unwarranted [18, 19].

The IC algorithm we used is a generic IC procedure for general sparse matrices taken from [7]. For reference convenience, it is reproduced in Algorithm 4.1, where we use the notations $M = (m_{i,j})$ and $A = (a_{i,j})$.

Algorithm 4.1 Procedure for incomplete Cholesky factorization.

1. $m_{1,1} = \sqrt{a_{1,1}}$.
2. For $i = 2$ to n
3. For $j = 1$ to $i - 1$
4. If $a_{i,j} = 0$ then $m_{i,j} = 0$ else
5. $m_{i,j} = \left(a_{i,j} - \sum_{k=1}^{j-1} m_{i,k} m_{j,k} \right) / m_{j,j}$
6. $m_{i,i} = \sqrt{a_{i,i} - \sum_{k=1}^{i-1} m_{i,k}^2}$

Since A is a Stieltjes matrix, Algorithm 4.1 is guaranteed to finish [12]. Algorithm 4.1 actually computes a lower triangular matrix M_L . The preconditioner is then taken as $M = M_L M_L^T$. For the current problem with a matrix of 5 diagonals, it is possible to have a more efficient implementation of PCG using the ideas of Eisenstat [6]. Since we are interested in using our code in more general situations, the Eisenstat implementation is not adopted.

Remarks. There are some aspects that should be considered when implementing a preconditioned CG method.

- Initial guess: at each time step $(n + 1)\Delta t$, the initial guess of $T_{i,j}^{n+1}$ for the PCG iteration is taken as the solution of the previous time step $T_{i,j}^n$. This is especially important when n is large and $T_{i,j}^{n+1}$ is approaching steady state. Since in the case of large t (and large n), $\max_{i,j} |T_{i,j}^{n+1} - T_{i,j}^n|$ is small.
- Stopping criterion and tolerance: an iteration scheme may have different stopping tolerance to terminate, corresponding to different stopping criterion. The most commonly used one is to measure the reduction rate of the residual in a certain norm, relative to the initial residual norm. This type of relative tolerances may not take advantage of a good initial guess, such as in the current case, if the stopping criterion is set too strict. On the other hand, a too liberal stopping criterion may result in an approximate solution that is not fully converged to the truncation accuracy. In our tests, the 2-norm residual reduction rate of 10^7 was set. More strict stopping tolerance could not yield smaller error.

- **Scaling matrix:** the coefficient matrix may have very large entries due to the appearance of Δx^2 and Δy^2 in the denominators of the representations of a, b, c and $F_{i,j}^n$. Since the y direction is at the microscale, $1/\Delta y^2$ can be quite large. It can produce very large values for the coefficients a, c and the right-hand side $F_{i,j}^n$. We experienced failure of constructing the IC preconditioner using the 32 bit arithmetic computation with large values of N , due to the overflow of data representation. This problem was not present when the 64 bit arithmetic was implemented. However, it is preferable to scale the linear system (the coefficient matrix and the right-hand side) by a factor $\Delta x^2 \Delta y^2$ so that all computation of coefficients and the right-hand side is performed with moderate size numbers. After they are computed, each equation is then scaled by a to make the main diagonal as unit.
- **Ordering of unknowns:** it is known that the ordering of the unknowns can affect the convergence rate of the PCG method with stationary iterative methods as preconditioners. We expect the ordering of the unknowns may affect our implementation since $|c|$ is usually much larger than $|b|$. In fact, if y axis is at the microscale, the natural (lexicographical) ordering will order the grid points following the x direction first, which is the strong direction. Studies on ordering effect on PCG preconditioned by stationary iterative methods in solving convection diffusion equation show that the ordering following the strong direction favors PCG convergence [2]. However, the ordering effect did not appear in our numerical tests, partly because the PCG method (preconditioned by IC) converges too fast to allow any significant convergence difference to be noticed.

The entire simulation process is formulated in Algorithm 4.2.

Algorithm 4.2 Simulation procedure for microscale heat transport equation.

1. *Given initial and boundary conditions, $\Delta t, \Delta x, \Delta y$ and time t_{stop}*
2. *Compute T^0 and θ^0*
3. *Compute coefficients a, b, c and construct the IC preconditioner M*
4. *For $n = 0, 1, \dots, t_{\text{stop}}/\Delta t$, do*
5. *Compute F^n from (36)*
6. *Solve $AT^{n+1} = F^n$ using the PCG method*
7. *Compute θ^{n+1} from (14)*
8. *End do*

5 Numerical Validations

Numerical experiments were conducted to validate the proposed discretization scheme and the iterative solution method. A model problem was constructed by setting $\alpha = 1$, $\tau_q = \frac{1}{\pi^2} + 10^2$ and $\tau_T = \frac{1}{\pi^2} + 10^{-6}$, on a rectangular domain $0 \leq x \leq 1, 0 \leq y \leq 10^{-4}$. The boundary and initial conditions were set to satisfy the exact solution as

$$T(x, y, t) = e^{-\pi^2 t} \sin(\pi x) \sin(10^4 \pi y).$$

Table 1: Maximum absolute errors in the computed solution with different N and Δt . ($L = 1, \varepsilon = 10^{-4}$).

$\Delta t \backslash N$	11	21	51	81	101	151	201
0.01	4.00(-7)	4.06(-7)	4.08(-7)	4.08(-7)	4.08(-7)	4.08(-7)	4.08(-7)
0.005	9.65(-8)	9.79(-8)	9.83(-8)	9.84(-8)	9.84(-8)	9.84(-8)	9.84(-8)
0.001	8.44(-10)	8.82(-10)	8.93(-10)	8.95(-10)	8.96(-10)	8.96(-10)	8.96(-10)

As mentioned previously, the PCG iteration was terminated when the 2-norm residual was reduced by a factor of 10^7 . The errors reported were the maximum absolute errors between the approximate solution and the exact solution as $\max_{i,j} |T_{i,j} - T_{i,j}^{n+1}|$ at $t = t_{\text{stop}}$. The code was written in standard Fortran 77 programming language and was run on an SGI Power Challenge workstation using 64 bit arithmetic.

We first computed a few simulations using different values of N and Δt for $t_{\text{stop}} = 1$. The maximum absolute errors of the simulations are listed in Table 1. We find that, in all cases, the errors are very small, but do not decrease when the spatial mesh is refined. (The slight increase of errors when N is large was likely caused by rounding errors.) These results are very interesting. In all tests, there is no oscillatory solution was computed for all different choices of $\Delta t, \Delta x^2$ and Δy^2 . This verifies the unconditional stability of the proposed finite difference scheme.

We also did experiments using different numbers of grid points in the x and y directions. In particular, we used $N_x = 2N_y$ in our tests and repeated the tests done for Table 1. The maximum absolute errors observed were almost the same as those reported in Table 1, due to the fact that the spatial mesh grid does not affect the final accuracy significantly for this test problem.

In order to understand why the finite difference scheme is insensitive to the refinement of spatial mesh grid, we plotted in Figure 1 the maximum absolute errors at each time step t with $N = 101$ and $N = 201$. We see that, although the maximum absolute errors for both $N = 101$ and $N = 201$ tend to become very small as t elapses, the maximum absolute errors are smaller with $N = 201$ than those with $N = 101$ when t is not very large. However, the error difference is only within the magnitude of 10^{-6} even for small t . This experiment demonstrates that finer spatial mesh does produce more accurate solution. Since the maximum absolute errors with all spatial meshes converge to zero as the exact solution converges to zero when t is large, they do not differ very much for large t . This explains the interesting but strange data shown in Table 1.

In order to show the correctness of the finite difference scheme, we conducted another set of experiments using $L = \varepsilon = 1$. This choice does not reflect the microscale effect of the test problem, but only shows that the finite difference scheme does react to the change in spatial mesh. The results in Table 2 show that, in general, smaller maximum absolute errors were obtained when the mesh is fine.

Figure 2 compares the (unpreconditioned) CG and PCG with respect to the number

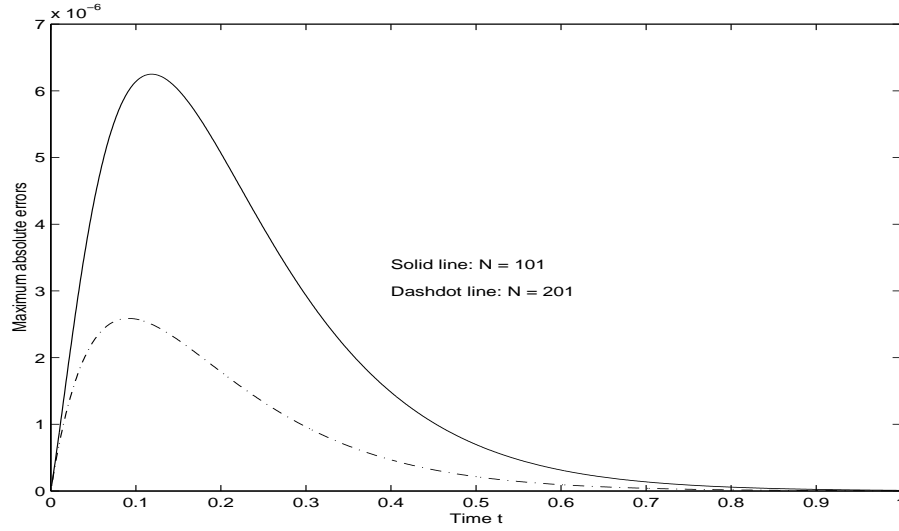


Figure 1: Comparison of maximum absolute errors at each time step t for $N = 101$ and $N = 201$. ($\Delta t = 0.001$, $t_{\text{stop}} = 1.0$, $L = 1$, $\varepsilon = 10^{-4}$).

Table 2: Maximum absolute errors in the computed solution with different N and Δt . ($L = \varepsilon = 1$).

$\Delta t \backslash N$	11	21	51	81	101	151	201
0.01	4.52(-3)	1.63(-3)	2.76(-4)	4.99(-7)	4.19(-7)	4.57(-7)	4.12(-7)
0.005	4.52(-3)	1.63(-3)	2.76(-4)	1.89(-7)	1.10(-7)	1.47(-7)	1.02(-7)
0.001	4.52(-3)	1.62(-3)	2.76(-4)	9.01(-8)	1.02(-8)	4.77(-8)	3.00(-9)

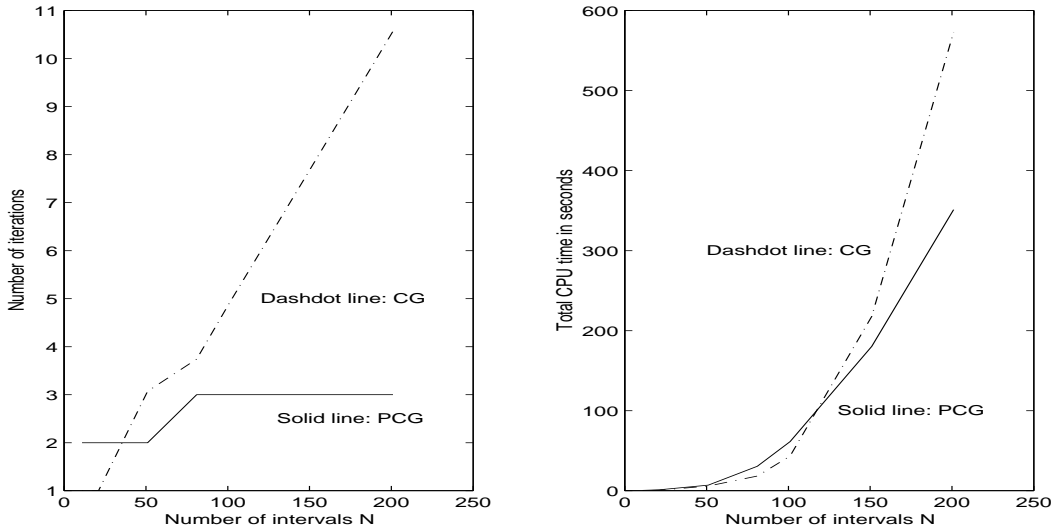


Figure 2: Comparison of iteration numbers and total CPU time in seconds between PCG and CG for solving the test problem with different N . ($\Delta t = 0.001$, $t_{\text{stop}} = 1.0$.)

of iterations (left figure) and the total CPU time in seconds (right figure), when different spatial mesh N was used. PCG took much less iterations than CG did to converge when N is large. The CPU time for PCG is also smaller when N is large. However, CG is shown to be more efficient when $N < 151$. This is because of the diagonal dominance of the coefficient matrix. It makes most simple iterative methods converge fast without any complicated implementation. However, for large size linear systems ($N \geq 151$), the fast convergence of PCG does demonstrate the usefulness of the IC preconditioner. Note that, for this particular problem, the preconditioning overhead can be reduced by using the Eisenstat implementation [6].

Figure 3 shows the numbers of iterations of CG and PCG at each time step with $N = 201$ and $\Delta t = 0.001$. The number of iterations of PCG is fixed at 3 throughout the entire simulation. The number of iterations of CG increases, statistically, as t becomes large. We can say that PCG is more robust than CG for solving large sparse linear systems.

6 Concluding Remarks

We have derived a two dimensional governing microscale heat transport equation and a few numerical techniques to solve the equation. We proposed a finite difference scheme to discretize the governing equation. The finite difference scheme has been proved to be unconditionally stable with respect to initial values. A preconditioned Conjugate Gradient method is used to solve the resulting sparse linear systems. The computational procedure proposed has been verified by the numerical experiments to be efficient and accurate.

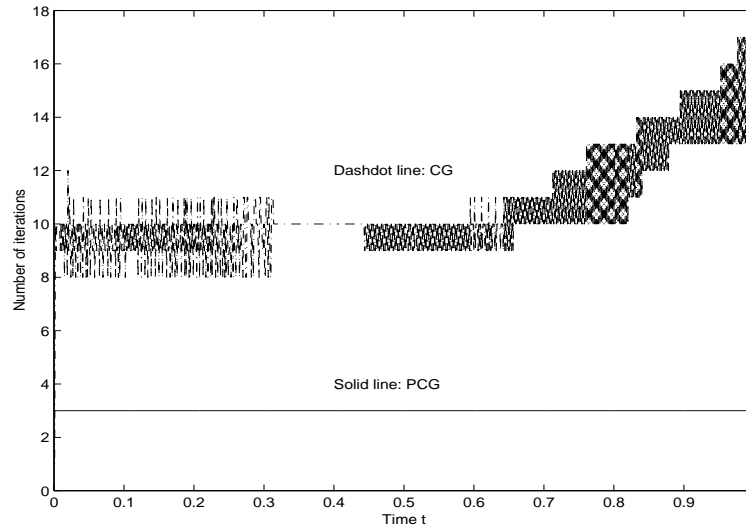


Figure 3: Comparison of iteration numbers between PCG and CG for solving the linear systems ($N = 201$) at each time step t . ($\Delta t = 0.001$, $t_{\text{stop}} = 1.0$.)

References

- [1] I. W. Byod. *Laser Processing of Thin Films and Microstructures*. Springer, New York, NY, 1989.
- [2] M. P. Chernesky. On preconditioned Krylov subspace methods for discrete convection-diffusion problems. *Numer. Methods Partial Differential Equations*, 13:321–330, 1997.
- [3] D. E. Chryssolouris. *Laser Machining, Theory and Practice*. Springer, New York, NY, 1991.
- [4] W. Dai. An unconditionally stable three level explicit difference scheme for the Schrödinger’s equation with a variable coefficient. *SIAM J. Numer. Anal.*, 29:174–181, 1992.
- [5] W. Dai and R. Nassar. A finite difference method for solving the heat transport equation at the microscale. *Numer. Methods Partial Differential Equations*, 15:697–708, 1999.
- [6] S. C. Eisenstat. Efficient implementation of a class of preconditioned conjugate gradient methods. *SIAM J. Sci. Statist. Comput.*, 2:1–4, 1981.
- [7] G. H. Golub and J. M. Ortega. *Scientific Computing: An Introduction with Parallel Computing*. Academic Press, Boston, MA, 1993.
- [8] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436, 1952.

- [9] D. D. Joseph and L. Preziosi. Heat waves. *Reviews Modern Phys.*, 61:41–73, 1989.
- [10] A. A. Joshi and A. Majumdar. Transient ballistic and diffusive phonon heat transport in thin films. *J. Appl. Phys.*, 74:31–39, 1993.
- [11] M. Lees. Alternating direction and semi-explicit difference methods for parabolic partial differential equations. *Numer. Math.*, 3:398–412, 1961.
- [12] J. A. Meijerink and H. A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comp.*, 31:148–162, 1977.
- [13] J. Narayan, V. P. Godbole, and G. W. White. Laser method for synthesis and processing of continuous diamond films on nondiamond substrates. *Science*, 52:416–418, 1991.
- [14] M. N. Özisik and D. Y. Tzou. On the wave theory in heat conduction. *ASME J. Heat Transfer*, 116:526–536, 1994.
- [15] D. Peaceman and H. Rachford. The numerical solution of elliptic and parabolic differential equations. *J. of SIAM*, 3:28–41, 1955.
- [16] T. Q. Qiu and C. L. Tien. Short-pulse laser heating on metals. *Int. J. Heat Mass Transfer*, 35(3):719–726, 1992.
- [17] T. Q. Qiu and C. L. Tien. Heat transfer mechanisms during short-pulse laser heating of metals. *ASME J. Heat Transfer*, 115:835–841, 1993.
- [18] Y. Saad and J. Zhang. BILUM: block versions of multielimination and multilevel ILU preconditioner for general sparse linear systems. *SIAM J. Sci. Comput.*, 20(6):2103–2121, 1999.
- [19] Y. Saad and J. Zhang. BILUTM: a domain-based multilevel block ILUT preconditioner for general sparse matrices. *SIAM J. Matrix Anal. Appl.*, 21(1):279–299, 1999.
- [20] D. Y. Tzou. Experimental support for the lagging behavior in heat propagation. *J. Thermophysics Heat Transfer*, 9(4):689–693, 1995.
- [21] D. Y. Tzou. The generalized lagging response in small-scale and high-rate heating. *Int. J. Heat Mass Transfer*, 38(17):3231–3240, 1995.
- [22] D. Y. Tzou. A unified field approach for heat conduction from micro to macro scales. *ASME J. Heat Transfer*, 117:8–16, 1995.
- [23] J. Zhang and J. J. Zhao. High accuracy stable numerical solution of 1D microscale heat transport equation. Technical Report No. 297-00, Department of Computer Science, University of Kentucky, Lexington, KY, 2000.