An Efficient One Dimensional Parabolic Equation Solver using Parallel Computing

Md. Rajibul Islam
Ibnu Sina Institute, Faculty of Science, University Technology Malaysia, Johor
md.rajibul.islam@gmail.com

Abstract

This paper will discuss the heat equation or as known as parabolic equation by Jacobi, Gauss Seidel and Alternating Direct Implicit (ADI) methods with the implementation of parallel computing on it. The numerical method is emphasized as platform to discretize the one dimensional heat equation. The result of three types of numerical methods will be presented graphically. The parallel algorithm is used to solve the one dimensional parabolic equation. Parallel Virtual Machine (PVM) is used in support of the communication among all microprocessors of Parallel Computing System. PVM is a software system that enables a collection of heterogeneous computers to be used as coherent and flexible concurrent computational resource. The numerical results and how fast of the convergence by Jacobi, Gauss-Seidel, and ADI will be evaluated, which are the main effort of this paper in order to fabricate an efficient One Dimensional Parabolic Equation Solver (ODPES). This new well-organized ODPES technique will enhance the research and analysis procedure of many engineering and mathematic fields.

Keywords: Parabolic equation, Jacobi, Gauss-Seidel, Alternating Direct Implicit (ADI) method, Parallel Computing

1. INTRODUCTION

It is abundantly clear that many important scientific problems are governed by partial differential equations according to [1-6, 14]. The difficulty in obtaining exact solution arises from the governing partial differential equations and the complexities of the geometrical configuration of physical problems [2, 3, 5]. For example, imagine a metal rod insulated along its length with no heat can escape for its surface. If the temperature along the rod is not constant, then heat conduction takes place. In such situations, the numerical method is used to obtain the numerical solutions [4]. These partial differential equations may have boundary value problems as well as initial value problems. This study will discuss the one-dimensional parabolic equation solved by three important methods, namely Alternating Direct Implicit, Jacobi Iteration, and Gauss-Seidel Methods. First, the parabolic equation will be written into the matrix form to ease the work. Then, the heat equation will run with parallel computing to provide the numerical solution. Finally, the speed of convergences of using the above numerical methods will be compared. In general, the transient particle diffusion or heat conduction is Partial Differential Equations (PDE) of the parabolic type [1, 5, 6]. The parabolic partial differential equations are normally used in such fields like molecular diffusion, heat transfer, nuclear reactor analysis, and fluid flow [8, 13, 15].

The parabolic partial differential equations represent time-dependent diffusion processes, therefore the t and x usually used as independent variables where t is time and x is the one-dimensional space coordinate.

Some examples of one-dimensional parabolic equations are

a) Transient neutron diffusion equation in one-space dimension [Hetric]:

$$\frac{1}{v}\frac{\partial}{\partial t}\phi(x,t) = D\frac{\partial^2 \phi}{\partial x^2} - \sum\nolimits_a \phi + v\sum\nolimits_f \phi + S$$

where φ is the neutron flux.

b) Convective transport of chemical species with dimension [Brodkey`]:

$$\frac{\partial}{\partial t}\phi = -\frac{\partial}{\partial x}u(x)\phi + D\frac{\partial^2}{\partial x^2}\phi$$

where φ is the density of the chemical species and u(x) is the flow velocity, and D is the diffusion constant.

The next sections of this paper will present the One Dimensional Parabolic Equation Solver (ODPES) as well as the alternating direct implicit methods, Gauss-Seidel methods and Jacobi methods and discretization of these numerical methods. Again the comparative study of the numerical solution of parabolic equation solved by alternating direct implicit methods, Gauss-Seidel methods and Jacobi methods and the speed evaluation of converges of the Alternating Direct Implicit (ADI), Gauss-Seidel and Jacobi methods is presented. The analysis of the performance of Parallel Virtual Machine (PVM) is elucidated in the result and discussion section.

The aim of this study is to solve the heat equation problem, which is always found in the engineering and scientific field by using the numerical methods while the analytical methods failed. This study will help to solve the heat equation problem with the numerical methods along with parallel computing efficiently.

A. Finite Difference Method

Finite Difference Method is a classical and straightforward way to solve numerically the partial difference equation [8, 10]. It consists of transforming the partial derivatives in difference equations over a small interval and the continuous domain of the state variables by a network or mesh of discrete points. The partial differential equation is converted into a set of finite difference equations so that it can be solved subject to the appropriated boundary conditions.

Assuming that *u* is function of the independent variables x and y, then divided the x-y plan in mesh points equal to δx = *h* and δy = *k*,
Evaluate *u* at point *P* by:

$$u_p = u(ih, jk) = u_{i,j}$$

The value of the second derivative at *P* could also be evaluated by:

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_p = \left(\frac{\partial^2 u}{\partial x^2}\right)_{i,j} \cong \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}$$

$$\left(\frac{\partial^2 u}{\partial y^2}\right)_p = \left(\frac{\partial^2 u}{\partial y^2}\right)_{i,j} \cong \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{k^2}$$

2. ONE DIMENSIONAL PARABOLIC EQUATION SOLVER (ODPES)

A. 1-D Parabolic equation

A parabolic equation is also called as heat equation or transient diffusion equation.

$$a\frac{\partial^2 u}{\partial x^2} + b\frac{\partial^2 u}{\partial x \partial y} + c\frac{\partial^2 u}{\partial y^2} = e$$

$$b^2 = 4ac$$

$$\frac{\partial u}{\partial t} = k\frac{\partial^2 u}{\partial x^2} \tag{1}$$

The parabolic equation describes the temperature distribution in a bar as a function of time. The two independent variables are time t and distance x along the bar with initial point at a and end at b.

It can rewrite the equation (1) in the following matrix form:

$$\begin{bmatrix} c & 1 & \cdots & \cdots & 0 \\ 1 & c & 1 & \cdots & 0 \\ 0 & 1 & c & 1 & \vdots \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \ddots & c \end{bmatrix} \begin{bmatrix} u_{1,j+1} \\ u_{2,j+1} \\ u_{3,j+1} \\ \vdots \\ u_{n-1,j+1} \end{bmatrix} = \begin{bmatrix} F_1 - u_0 \\ F_2 \\ F_3 \\ \vdots \\ F_{n-1} - u_n \end{bmatrix} \tag{2}$$

Where u_0 and u_n are known temperatures at the end points of the bar [a, b], and the length of the bar has been divided into n equal segments with $\frac{b-a}{n}$.

B. Jacobi Iteration Method

In this method, the values of the guess vector are repeatedly used in the equations on the right hand side until a completely new solution vector is formed [7].

The values of the solution vector for the next iteration are ready only after all the equations on the right hand side have been evaluated with the old values in the guess vector. The procedure is continued until relative error between two consecutive vector norms become very small.

The Jacobi iteration method and the matrix vector products can be written in the element by element form,

$$u_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij} u_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} u_j^{(k)}\right]$$

(3)

where k=0, 1, 2,…

The first term in column represents the matrix *b,* the second term is from *L*x and the third term is the contribution from *U*x. Each estimate $u_i^{(k+1)}$ is determined solely in terms of the positive of the previous iterative $u^{(k)}$.

C. Gauss-Seidel Iteration Method

In equation (3), the improved estimate $u_1^{(k+1)}$ is evaluated in the first equation. Then, replace the value $u_1^{(k)}$ in equation (3) to *n* by this improved estimate for the first element. Similarly, once $u_2^{(k+1)}$ is evaluated from the second equation, it can be used to replace the value $u_2^{(k)}$ in third equation to n.

The update formula is known as the Gauss-Seidel iteration method. One expanding the matrix vector products, it can be written in the element by element form

$$u_i^{(k+1)} = \frac{1}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}u_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}u_j^{(k)}\right]$$

K= 0, 1, 2,………

The first term in column represents the vector *b*, the second term is the contribution from *Lu* and the third term is the contribution from *Uu*. Gauss-Seidel iteration is a successive correction method, whereas Jacobi iteration is a simultaneous correction method. Convergence is assured in Gauss-Seidel method if matrix A is diagonally dominant and positive definite. If it is not a diagonally dominant form, it should be converted to a diagonally dominant form by row interchanges, before starting the Gauss-Seidel iterations.

D. Alternating Direct Implicit Method, ADI

The first of these were developed by Peaceman and Rachford (PRADI for short), is related to procedure developed by Douglas for solving the diffusion equation. Douglas and Rachford presented a method similar to that of PRADI characterized by its ease of generalization to three dimensions [12]. Birkhoff, Varga, and Young summarize the state of knowledge to 1962 in a long survey paper complete with a lengthy series of computations and numerous references. Tateyama, Umoto, and Hayashi present a variant of the classical PRADI. Their double interlocking variant proceeds on every other line using old values and on the remainder using the new values.

Consider the equation

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} + gu = f(x, y) \qquad (4)$$

where the constant $g \geq 0$.

The Peaceman-Rachford A.D.I iteration is defined by the equations,

$$-u_{i-1,j}^{(n+1)}+(2+\frac{1}{2}h^2g+\rho)u_{i,j}^{(n+1)}-u_{i+1,j}^{(n+1)}=$$

$$u_{i,j-1}^{(n)}-(2+\frac{1}{2}h^2g-\rho)u_{i,j}^{(n)}+u_{i,j+1}^{(n)}+h^2f_{i,j}$$

and

$$-u_{i,j-1}^{(n+2)}+(2+\frac{1}{2}h^2g+\rho)u_{i,j}^{(n+2)}-u_{i,j+1}^{(n+2)}=$$

$$u_{i-1,j}^{(n+1)}-(2+\frac{1}{2}h^2g-\rho)u_{i,j}^{(n+1)}+u_{i+1,j}^{(n+1)}+h^2f_{i,j}$$

Note that the equation can be transform into matrix form as follows:

$$\begin{bmatrix}\alpha & -1 & & & \\ -1 & \alpha & -1 & & \\ & & \ddots & \ddots & \\ & & & & -1 \\ & & & -1 & \alpha\end{bmatrix}\begin{bmatrix}u_{i,j}\\ u_{i,j+1}\\ \vdots\\ \vdots\\ u_{i,j}\end{bmatrix}^{n+2}=$$

$$\begin{bmatrix}\beta & 1 & & & \\ 1 & \beta & 1 & & \\ & \ddots & \ddots & & \\ & & \ddots & 1 \\ & & 1 & \beta\end{bmatrix}\begin{bmatrix}u_{i,j}\\ u_{i+1,j}\\ \vdots\\ \vdots\\ u_{i,j}\end{bmatrix}^{n+1}+\begin{bmatrix}h^2f_{i,j}\\ \vdots\\ \vdots\\ \vdots\\ \vdots\end{bmatrix}$$

3. DISCRETIZATION OF HEAT EQUATION

The approach to solve parabolic partial differential equations by numerical methods is to replace the partial derivatives by using the following methods, it also named discretization.

A. Discretization using Alternating Direct Implicit Method (ADI)

First the equation must be approximated to

$$u_{i,j}(t)=\frac{C}{h^2}\left[u_{i-1,j}(t)-2u_{i,j}(t)+u_{i+1,j}(t)\right] \tag{5}$$

Equation (5) may be used for *i*=1, 2, 3, …,(*n*-1) and *j*=1, 2, 3,…,(*n*-1), giving a set of (m-1)(n-1) simultaneous differential equations for the unknown $u_{i,j}(t)$. The problem, that remains inconsiderable to solve by these equations. Replace $u_{i,j}(t)$ in (5) by approximation, an explicit finite difference formula will be obtained.

$$\frac{1}{k}\left[u_{i,j}(t+k)-u_{i,j}(t)\right] \tag{6}$$

and an implicit finite difference formula

$$\frac{1}{k}\left[u_{i,j}(t)-u_{i,j}(t-k)\right] \tag{7}$$

When combine two of them, the following equation will be obtained

$$\frac{2}{k}\left[u_{i,j}(t+\frac{1}{2}k)-u_{i,j}(t)\right]=$$

$$\frac{C}{h^2}\left[u_{i-1,j}(t+\frac{1}{2}k)-2u_{i,j}(t+\frac{1}{2}k)+u_{i+1,j}(t+\frac{1}{2}k)\right] \quad (8)$$

B. Disretization using Jacobi Method

Consider the Crank-Nicholson method for deriving a numerical solution of

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2} \quad (9)$$

as the following

$$\frac{u_{i,j+1}-u_{i,j}}{k}=$$

$$\frac{1}{2}\left\{\frac{u_{i-1,j+1}-2u_{i,j+1}+u_{i+1,j+1}}{h^2}+\frac{u_{i-1,j}-2u_{i,j}+u_{i+1,j}}{h^2}\right\} \quad (10)$$

An elementary modification to (10), gives

$$u_i^{(n+1)}=\frac{1}{2}r\{u_{i-1}^{(n)}-2u_i^{(n)}+u_{i+1}^{(n)}\}+b_i \quad (11)$$

Formula (11) which calculates the (n+1) the iterates exclusively in terms of the *n*th iterates, is called the Jacobi iteration corresponding to equations (10).

C. Discretization using Gauss-Seidel Method

The discretization of Gauss-Seidel method is very similar to Jacobi method, the only difference between them is the replacement of $u^{(n)}$ by $u^{(n+1)}$, then the equation (11) gives

$$u_i^{(n+1)}=\frac{1}{2}r(u_{i-1}^{(n+1)}-2u_i^{(n+1)}+u_{i+1}^{(n)})+b_i \quad (12)$$

D. Case Study: One Dimensional Parabolic Equation Problem

The temperature distribution in a bar can be described by the one dimensional parabolic equation with k=1, and boundary conditions u(0,t) = 0, and u(1,t) = 1 and initial condition u(x,0)=4x, for 0≤ x ≤0.5 and u(x,0) =3-2x for 0.5 ≤ x ≤1. Determine how the temperature changes in time using Δx=0.02 and Δt=0.02.

Solution:

To solve this problem, it may apply equation (11) in section 2, where n= (1-0)/0.02=50; u(0)=0, u(1)=1.

Then, the F_i and $u_{i,o}$ will be calculated by using Excel as following.

Table 1: Computed F_i and $u_{i,o}$

i	$u_{i,o}$	F_i
0	0	0
1	0.08	-0.0032
2	0.16	-0.0064
3	0.24	-0.0096
4	0.32	-0.0128
5	0.4	-0.016
6	0.48	-0.0192
7	0.56	-0.0224
8	0.64	-0.0256
9	0.72	-0.0288
10	0.8	-0.032
11	0.88	-0.0352
12	0.96	-0.0384
13	1.04	-0.0416
14	1.12	-0.0448
15	1.2	-0.048
16	1.28	-0.0512
17	1.36	-0.0544
18	1.44	-0.0576
19	1.52	-0.0608
20	1.6	-0.064

4. PARALLEL COMPUTING

A parallel computing system is a computer with more than one processor for parallel processing. Some parallel computers are just regular workstations that have more than one processor in them but others are giant single computers with many processors. These are generally referred to as supercomputers and others are networks of individual computers. Parallel computers can run some types of programs far faster than traditional single processor computers. Scientific applications are already using parallel computation as a method for solving problems, for example the heat equation. Parallel computers are normally used in computationally intensive applications like climate modeling, finite element analysis and digital signal processing. The key benefit of parallel computing is to solve large and complex problems fast [11]. Other benefits include cost savings, taking advantage of non-local resources, computing resources instead of a supercomputer, and overcoming memory constraints on single computer.

Parallel Virtual Machine (PVM) is recently the most popular communication protocol for implementing distributed and parallel applications. PVM is an on-going project and is freely distributed. The PVM software provides a framework to develop parallel programs on a network of heterogeneous machines. PVM handles message routing, data conversion and task scheduling.

PVM has facilities to do many additional things, including speed up message passing by special specifying the packing methods, encoding and routing. Besides, PVM also can do purpose calls to speed up homogenous data messages and calls the group function including broadcasting, reducing, scattering, and gathering.

A. Performance Analysis

Performance analysis is one of the important part in parallel programming. The reason for writing parallel programs is to solve a problem faster than a serial computer, and solve a larger problem as well. Measuring the performance of a parallel program is a way to assess how well and how efficient the efforts of parallel computing used to run a program.

The performance analysis of PVM will be discussed in this subsection. Various methods are used to measure the performance of a certain parallel program. The performance analysis will be measured by the execution time, speedup, efficiency, effectiveness, and temporal performance. The measurement are all related to each other, but highlight different components of the dynamics at work during the execution of the parallel program. The following subtopic will show the calculation and the graph of the performance analysis.

B. Execution Time

The most visible and easily recorded measurement of performance is the execution time. Execution time is the time to execute a program using N processor. The following is the execution time of using 35 processor to execute a program by parallel computing. By measuring how long the parallel program needs to run to solve the problem, it can directly measure its effectiveness.

Table 2: Execution Time

Number of processors	Time (Micro Second)		
	Jacobi	**GS**	**ADI**
1	42228568	23483497	5114789
5	25442030	12186957	2526692
10	18534848	8799795	1810049
15	11764352	6169910	1213118
20	9593550	4785296	984231
25	8212153	3926748	827690
30	6541995	3346007	704400
35	5735209	3022810	622561

From the above graph, it can be easily seen that execution time of Gauss-Seidel, ADI, and Jacobi are decreased with the increasing number of processor. Besides, the execution time using by Jacobi and GS are more than ADI.

C. Speedup

Speedup also called parallel speedup. A straightforward measurement of the speedup would be the ratio of the execution time on a single processor to that on N processors. Speedup describes the scalability of the system as the number of processors is increased. The ideal speedup is n when using *N* processors, such as when the computations can be divided into equal duration processes with each process running on one processor.

Table 3: Speedup

Number of processors	Time (Micro Second)		
	Jacobi	**GS**	**ADI**
1	1	1	1
5	1.65979	1.92693	2.02430
10	2.27833	2.66864	2.82577
15	3.58954	3.80613	4.21623
20	4.40177	4.90742	5.19674
25	5.14220	5.98039	6.17959
30	6.45500	7.01836	7.26120
35	7.36304	7.76876	8.21572

Table 3 shows that the speedup graph is increasing as the number of processor, N increases. According to Amdahl's Law, the speedup increases with the number of processors increase up to certain level. Therefore, the parallel computing is proved to be suitable for large sparse matrix problem, such as the heat equation problem and ADI provides better result with the increase of processor quantity than Gauss-Seidel and Jacobi methods.

D. Efficiency

Efficiency is the speedup divided by the number of processors. It is a value between zero and one, estimating how well-utilized the processors are in solving the problem, compared to how much effort is wasted in communication and synchronization. The efficiency is defined as the average contribution of the processors towards the global computation.

Table 4: Efficiency

Number of processors	Time (Micro Second)		
	Jacobi	**GS**	**ADI**
1	1	1	1
5	0.33195911	0.38538738	0.40486051
10	0.22783337	0.26686414	0.28257738
15	0.23930242	0.25374219	0.28108225
20	0.22008833	0.24537141	0.25983682
25	0.20568817	0.23921573	0.2471838
30	0.21516662	0.23394549	0.24203999
35	0.21037255	0.22196468	0.23473496

Table 4 indicates the efficiency decreases with the increasing of number of processors, *N*. This is because of the imbalance workload distributed among the different processor. Providing effectual workload balancing to large scale simulations is still an open issue. This will be our further research in order to increase the efficiency of parallel performance estimation.

E. Effectiveness

Effectiveness is used to calculate the speedup and the efficiency. Effectiveness is the speedup divided by number of processors execution time. It is defined in Table 5.

Table 5: Effectiveness

Number of processors	Time (Micro Second)		
	Jacobi	GS	ADI
1	2.36807E-08	4.25831E-08	1.95511E-07
5	1.30477E-08	3.16229E-08	1.60233E-07
10	1.22922E-08	3.03262E-08	1.56116E-07
15	2.03413E-08	4.11258E-08	2.31702E-07
20	2.29413E-08	5.12761E-08	2.64E-07
25	2.50468E-08	6.09196E-08	2.98643E-07
30	3.28901E-08	6.99178E-08	3.43612E-07
35	3.66809E-08	7.34299E-08	3.77047E-07

Table 5 shows that the effectiveness increases as the number of processors increase. From the formula, it can be seen that the effectiveness depends on the speedup, therefore the effectiveness will increase when the speedup increase.

F. Temporal Performance

Temporal performance is parameter to measure the performance of a parallel algorithm .It is defined as

Table 6: Temporal Performance

Number of processors	Time (Micro Second)		
	Jacobi	GS	ADI
1	2.36807E-08	4.25831E-08	1.95511E-07
5	3.9305E-08	8.20549E-08	3.95774E-07
10	5.39524E-08	1.13639E-07	5.52471E-07
15	8.50026E-08	1.62077E-07	8.24322E-07
20	1.04237E-07	2.08973E-07	1.01602E-06
25	1.21771E-07	2.54664E-07	1.20818E-06
30	1.52859E-07	2.98864E-07	1.41965E-06
35	1.74362E-07	3.30818E-07	1.60627E-06

Table 6 shows that, the temporal performance of Jacobi, GS, and ADI methods are increasing with the increasing number of processors. Among of the three above methods, ADI method is perform better than Jacobi and GS method since the execution time of ADI method is less than Jacobi and GS methods. This means that the parallel algorithms which are using least time to execute, therefore the algorithms are the best.
Table 6 indicates that the temporal performance is increase with the increasing number of processors. From the graph, it can be seen that ADI is perform better than GS and Jacobi.

The analysis from the aspect of speedup, efficiency, effectiveness, and the temporal performance showed that the performance of PVM is improved as the number of processors increase. The result of the performance analysis have proved that to measure the execution time, speedup, effectiveness, temporal performance, and efficiency using parallel algorithm is significantly better than the sequential. Therefore, the parallel computing is becoming more popular because it saves a lot of time in solving large sparse problem. There are a lot of type of parallel computing such as Message Passing Interface (MPI), Application Programming Interface (API), and Parallel Virtual Machine (PVM). PVM is a software environment for heterogeneous distributed computing. It allows a user to create and access a parallel computing system made from a collection of distributed processors, and treat the resulting system as a single virtual. This machine has provided a new capability to the scientists and engineers, and they have been successfully used by the scientists and engineers even though with varying degrees of success.

5. CONVERGENCE OF NUMERICAL METHODS

This paper is concerned with the convergence of the solution of one dimensional parabolic equation. The Jacobi and Gauss-Seidel methods are said to be convergent if A is a diagonally dominant matrix, although it may converge even if A is not diagonally dominant matrix. Meanwhile, the convergent of ADI method depends on a suitable choice of iteration parameter. Generally, each of the three iterative methods described above can be written as

$$u^{(n+1)} = Au^{(n)} + e$$

where A is the iteration matrix and b is a column matrix of known values. The equations was derived from the original equations by rearranging them into the form

$$u = Au + b$$

The error, $e^{(n)}$ in the n^{th} approximation to the exact solution is defined by

$$e^{(n+1)} = Ae^{(n)}$$

Therefore,

$$e^{(n)} = Ae^{(n-1)} = A^2 e^{(n-2)} = ... = A^n e^{(0)}$$

The sequence of iterative values $u^{(1)}, u^{(2)}, ..., u^{(n)}$ will converge to u as n tends to infinity if

$$e^{(n)} = 0$$

As a corollary to this result, a sufficient condition for convergence is that ‖A‖<1.

A. Conditions on the Convergence of Jacobi Method

The Jacobi iterative method can be written as

$$u^{(k+1)} = D^{-1}(L + D)u^{(k)} + D^{-1}b \qquad \text{k=0,1,2,.......}$$

Let

$$u^{(k+1)} = T_j u^{(k)} + u$$

To assure the convergence of the Jacobi methods, it must has the spectral radius, $\rho(T_j) < 1$. Since the calculation of $\rho(T_j)$ is often very complicated, one should be satisfied with the sufficient conditions like $\|T_j\| < 1$. Now, if

$$T_j = (t_{ij}), 1 \le i, j \le n,$$

Then by the definition of $\|\bullet\|_\infty - norm, \|T_j\| \leq 1$
if and only if

$$\sum_{j=1}^{n} t_{ij} < 1, \forall i = 1,2,.....,n$$

and since $T_j = D^{-1}(L+U)$,this implies that

$$\sum_{j=1}^{n} |l_{ij} + u_{ij}| < |d_{ij}| \forall i$$

Therefore, by the definition of *D, L*, and *U* the sufficient condition for the convergence of the Jacobi method to solve *Ax*=*b* will be written as

$$\sum_{j \neq 1}^{n} |a_{ij}| < |a_{ii}|, \forall i = 1,2,.....,n$$

A sufficient condition for the convergence of Jacobi method is that the matrix *A* of the linear system *Ax* =*b* is diagonally dominant.

B. Conditions on the Convergence of Gauss-Seidel Method

Gauss-Seidel method will converge if

$\sum_{j=1}^{n} |a_{ij}| < |a_{ii}|$ for $i = 1,2,.....,n$. The above condition will ensure the convergence if the matrix *A* is diagonally dominant.

6. NUMERICAL SOLUTION COMPARISON

From the output of the C programme, it can be easily seen that the error of using ADI method is smaller than the tolerance, in twenty iterations. Meanwhile the error of using Gauss-Seidel and Jacobi methods are smaller than the tolerance in thirty seven iteration and fifty four iteration, respectively. Since the error of using ADI method is smaller than the error of Gauss-Seidel and Jacobi methods, therefore it can be concluded that the convergence of ADI method is faster than Gauss-Seidel and Jacobi methods.

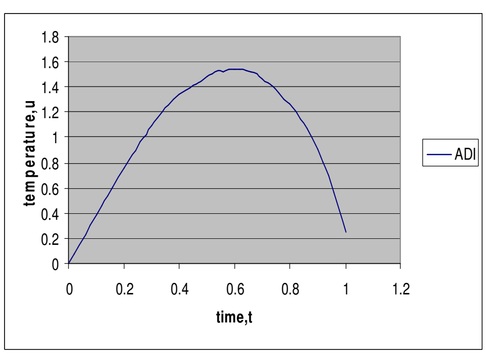

Figure 1: Temperature behavior of ADI method

Figure 1 shows that the temperature increases with the time. The temperature decreased at U[24]=1.4063 and U[25] =1.3770 as showed in the output. Then, the temperature increased again at U[26]=1.4436. Therefore, the graph is not a very smooth curve. From the output, the error of ADI method is the smallest that is 0.0008061 if compared with the Jacobi method, that is 0.000983, and Gauss-Seidel method, 0.0009988.

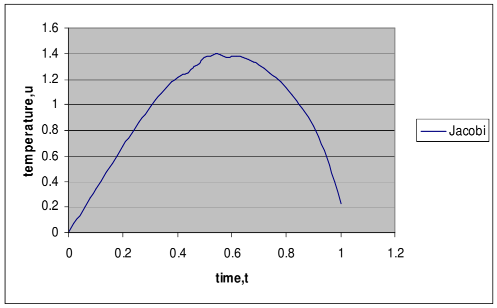

Figure 2: Temperature behavior of Jacobi method

Figure 2 is slightly same with figure 1 solved by using ADI method but the value of U[24]=0.9580, U[25]=0.9235, and U[26]=0.984243. From the output, it can be seen that the error of using Jacobi method is larger than ADI method.

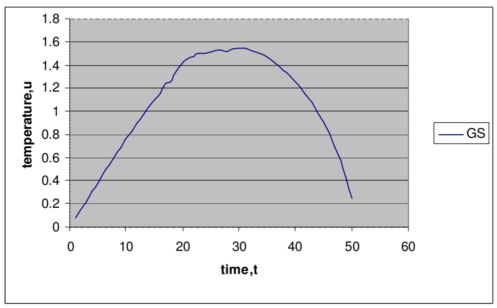

Figure 3: Temperature behavior of Gauss-Seidel method

The graph above indicates that the temperature increased with time, the value of U[24]=1.0759, U[25]=1.1409,and U[26]=1.1872 and the error of using the Gauss-Seidel method is larger than the ADI method and the Jacobi method (see Figure 3).

7. CONCLUSION

Numerical techniques in solving scientific and engineering problems are growing importance, and the subject has become an essential part of the training of applied mathematicians, engineers and scientists.

The reason is numerical methods can provide the solution while the ordinary analytical methods fail [9]. Numerical methods have almost unlimited breadth of application. Other reason for lively interest in numerical procedures is their interrelation with digital computers [10]. Besides, parallel computing is a good tool to solve a large scale problem especially the numerical problem. Parallel computing is time saving comparatively with the sequential computing as well as other programming.

In this study, the ADI method performs with much higher computational efficiency for a large class of problems. ADI method is more suitable for vector processors of a super computer. Meanwhile, the Gauss-Seidel method is converges almost twice faster than Jacobi methods although the error of using Gauss-Seidel method is larger than Jacobi method. It is because of the new value u_i of Gauss-Seidel method immediately replacing the old value of u_i in the vector $u_i^{(n)}$. However, the Gauss-Seidel method is converges slower if compared with ADI method.

Therefore, the ADI method is suggested to use to solve the one-dimensional parabolic equation among the three numerical methods although its equations is considered more complex compare to Jacobi and Gauss-Seidel methods.

8. REFERENCES

[1] Norma Alias, Md. Rajibul Islam, Nur Syazana Rosly, "A Dynamic PDE Solver for Breasts' Cancerous Cell Visualization on Distributed Parallel Computing Systems", in *Proc. The 8th International Conference on Advances in Computer Science and Engineering (ACSE 2009)*, Phuket, Thailand, pp. 138-143, Mar., 2009.

[2] Alias, N., Sahimi, M.S., and Abdullah, A.R., "The AGEB Algorithm for Solving the Heat Equation in Two Space Dimensions and Its Parallelization on a Distributed Memory Machine", Proceedings of the *10th European PVM/ MPI User's Group Meeting: Recent Advances In Parallel Virtual Machine and Message Passing Interface*, Vol. 7, pp. 214–221, 2003.

[3] Alias, N., Sahimi, M.S., Abdullah, A.R., "Parallel Strategies for the Iterative Alternating Decomposition Explicit Interpolation-Conjugate Gradient Method In solving Heat Conductor Equation on a Distributed Parallel Computer Systems". Proceedings of *the 3rd International Conference on Numerical Analysis in Engineering*, Vol. 3, pp. 31-38, 2003.

[4] Tan, Y.S., *Parallel Algorithm for One-dimensional Finite Element Discretization of Crack Propagation*. Universiti Teknologi Malaysia. 2008. MSc thesis.

[5] Norma Alias, Rosdiana Shahril, Md. Rajibul Islam, Noriza Satam, Roziha Darwis, "3D parallel algorithm parabolic equation for simulation of the laser glass cutting using parallel computing platform", *The Pacific Rim Applications and Grid Middleware Assembly (PRAGMA15)*, Penang, Malaysia. Oct., 2008. (Poster presentation).

[6] Evans D.J, Sukon K.S, "The Alternating Group Explicit (AGE) Iterative Method for Variable coefficient Parabolic Equations", *Intern. J. Computer Math.* Vol 59, pp 107-121, 1995.

[7] Scraton, R.E., *Further Numerical Methods in BASIC*, Australia, Edward Arnold (Publishers) Ltd., 1987.

[8] Nakamura, Shoichiro, *Applied Numerical Methods In C*, United State of America PTR Prentice Hall Inc., 1993.

[9] B. Bhat, Rama, Chakraverty, Snehasnish, *United Kingdom Numerical Analysis in Engineering*, Alpha Science International Ltd., 2004.

[10] SMITH, G.D., *Numerical Solution of Partial Differential Equations*, Oxford Universities Press, 1965.

[11] Nakumura, Shoichiro, *Applied Numerical Methods with Software*, United State of America PTR, Prentice Hall. Inc., 1991.

[12] Douglas. J, Burden, Faires Richard, *Numerical Methods* (Third Edition), Prentice Hall. Inc., 2003.

[13] SMITH, G.D, *United Kingdom, and Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Oxford Universities Press, 1985.

[14] Norma Alias and Md. Rajibul Islam, "A Review of the Parallel Algorithms for Solving Multidimensional PDE Problems", *Journal of Applied Sciences*. Vol. 10, No., 19, pp. 2187-2197, 2010.

[15] Md. Rajibul Islam, Norma Alias, "A Case Study: 2D Vs 3D Partial Differential Equation toward Tumor Cell Detection on Multi-Core Parallel Computing Atmosphere", *International Journal for the Advancement of Science & Arts (IJASA)*, Vol. 1, No. 1, pp. 25-34, 2010.