SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING

MAE 6263 Course Project I

Romit Maulik
April 13, 2016

1 INTRODUCTION

The purpose of this report is to document the numerical solution of the 2-D parabolic heat equation on a square domain of side 1 unit. Through the course of this document, iterative numerical schemes shall be devised and the results of numerical experiments shall be analysed with the view of understanding the basic concepts of finite difference algorithms such as consistency, stability and different solution methods.

1.1 PROBLEM STATEMENT

The 2-D Parabolic heat equation can be defined for a Cartesian coordinate system as:

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}, \tag{1.1}$$

for a square domain (Ω) of unit side length and with initial conditions given by-

$$T(x, y) = 0 \text{ for } 0 < x < 1, \;\; 0 < y < 1, \tag{1.2}$$

and constant boundary conditions given by

$$T(x, y) = x + y \text{ on } d\Omega \tag{1.3}$$

2 Code development

2.1 Development of iterative schemes

2.1.1 Explicit iterative scheme

The partial differential equation given by Eq. (1.1) may be discretized by utilizing a forward time and central space discretization about a discrete field variable given by $T^n_{i,j}$, i.e. the solution of the PDE at a certain node in the discretized domain (i,j) known at time n. This may be described as -

$$\frac{T^{n+1}_{i,j} - T^n_{i,j}}{\Delta t} = \frac{T^n_{i+1,j} + T^n_{i-1,j} - 2T^n_{i,j}}{\Delta x^2} + \frac{T^n_{i,j+1} + T^n_{i,j-1} - 2T^n_{i,j}}{\Delta y^2}. \tag{2.1}$$

On reorganization of terms we can obtain an algebraic equation for the solution at a node for its new time step provided the solution is known at the previous time step -

$$T^{n+1}_{i,j} = \gamma(T^n_{i+1,j} + T^n_{i-1,j} - 2T^n_{i,j}) + T^n_{i,j}, \tag{2.2}$$

where γ is given by $\Delta t/\Delta x^2$ and we have assumed $\Delta x = \Delta y$. The pseudocode for an explicit method is given by -

Algorithm 1 Explicit solver for 2D heat equation

Require: $\gamma \leq 0.5$ & T^n_{ij}
  while $j > 0$ & $j < N$ **do**
    while $i > 0$ & $i < N$ **do**
      $T^{n+1}_{ij} = \gamma(T^n_{i+1,j} + T^n_{i-1,j} - 2T^n_{i,j}) + T^n_{i,j}$
      $i \leftarrow i+1$
    end while
    $j \leftarrow j+1$
  end while

2.1.2 Implicit iterative scheme

Similar to the procedure used to develop the explicit iteration scheme, a backward time and central space discretization can be undertaken at the $n+1$ timestep solution given by $T^{n+1}_{i,j}$ to obtain a system of equations which need to be solved simultaneously. This is explained as follows. On applying the discretization we get -

$$\frac{T^{n+1}_{i,j} - T^n_{i,j}}{\Delta t} = \frac{T^{n+1}_{i+1,j} + T^{n+1}_{i-1,j} - 2T^{n+1}_{i,j}}{\Delta x^2} + \frac{T^{n+1}_{i,j+1} + T^{n+1}_{i,j-1} - 2T^{n+1}_{i,j}}{\Delta x^2} \tag{2.3}$$

and rearrangement gives us -

$$(1+4\gamma)T^{n+1}_{i,j} - \gamma T^{n+1}_{i+1,j} - \gamma T^{n+1}_{i-1,j} - \gamma T^{n+1}_{i,j+1} - \gamma T^{n+1}_{i,j-1} = T^n_{i,j}. \tag{2.4}$$

It must be noted here that the right hand side of the above equation may be supplemented with boundary terms. A standard node numbering scheme is used to develop a solution vector with $(N-1)^2$ elements and the reader is directed to the textbook by Moin (2010) for a detailed description. A block tridiagonal matrix is thus developed which has to be iteratively inversed at each timestep -

$$\begin{bmatrix} B & C & & & \\ A & B & C & & \\ & . & . & . & \\ & & . & . & . \\ & & & A & B \end{bmatrix}, \tag{2.5}$$

with B given by -

$$B = \begin{bmatrix} 1+4\gamma & -\gamma & & & \\ -\gamma & 1+4\gamma & -\gamma & & \\ & . & . & . & \\ & & . & . & . \\ & & & -\gamma & 1+4\gamma \end{bmatrix}, \tag{2.6}$$

and A, C given by -

$$A, C = \begin{bmatrix} -\gamma & & & & \\ & -\gamma & & & \\ & & . & & \\ & & & . & \\ & & & & -\gamma \end{bmatrix}. \tag{2.7}$$

A final solution at each timestep is obtained implicitly by the inversion of a block tridiagonal matrix with dimensions $(N-1)^2$. If our system of equations is represented as -

$$\overline{\overline{A}}\overline{x} = \overline{b} \tag{2.8}$$

where A is our block tridiagonal matrix that needs to be inverted and $\overline{b}$ is the solution vector at the previous timestep. The unknown vector $\overline{x}$ represents the solution at the end of the current timestep being iterated for. The Gauss Seidel iterative method can be represented by a forward substitution operation given by -

$$x_i^{k+1} = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{k+1} - \sum_{j>i} a_{ij} x_j^k \right), i, j = 1, 2, ..., n. \tag{2.9}$$

The forward substitution operation is terminated when the successive results of iterating on the solution vector are deemed to be converged. The convergence criteria is set up in the sense of an L1 norm less than some threshold tolerance given by-

$$\sum_i^N \frac{|x_i^{k+1} - x_i^k|}{N} < 10^{-4}. \tag{2.10}$$

A pseudocode for this procedure can be given by-

Algorithm 2 Implicit solver for 2D heat equation

Require: $\overline{\overline{A}}, \overline{b}$ & initial guess $\bar{x}$

```
repeat
  for i = 1 to (N−1)^2 do
    σ_1 ← 0, σ_2 ← 0
    for j = 1 to (N−1)^2 do
      if j > i then then
        σ_1 ← σ + A_ij x_j^k
      else if j < i then then
        σ_2 ← σ + A_ij x_j^(k+1)
      end if
    end for
    x_i ← (1/A_ii)(b_i − σ_1 − σ_2)
  end for
until Check for convergence
```

2.2 Von Neumann stability analysis

2.2.1 Explicit method

The 2-D partial differential equation for the heat equation is given by -

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}. \tag{2.11}$$

We can decompose the computed solution as -

$$T_{i,j}^n = \overline{T_{i,j}}^n + \epsilon_{i,j}^n \tag{2.12}$$

and substitute it into our modified differential equation in Eq. (2.1) to give-

$$\begin{aligned} \frac{\overline{T_{i,j}}^{n+1} - \overline{T_{i,j}}^n}{\Delta t} + \frac{\epsilon_{i,j}^{n+1} - \epsilon_{i,j}^n}{\Delta t} &= \frac{\overline{T_{i+1,j}}^n + \overline{T_{i-1,j}}^n - 2\overline{T_{i,j}}^n}{\Delta x^2} + \frac{\overline{T_{i,j+1}}^n + \overline{T_{i,j+1}}^n - 2\overline{T_{i,j}}^n}{\Delta y^2} \\ &+ \frac{\epsilon_{i+1,j}^n + \epsilon_{i-1,j}^n - 2\epsilon_{i,j}^n}{\Delta x^2} + \frac{\epsilon_{i,j+1}^n + \epsilon_{i,j-1}^n - 2\epsilon_{i,j}^n}{\Delta y^2}, \end{aligned} \tag{2.13}$$

which then simplifies to

$$\frac{\epsilon_{i,j}^{n+1} - \epsilon_{i,j}^n}{\Delta t} = \frac{\epsilon_{i+1,j}^n + \epsilon_{i-1,j}^n - 2\epsilon_{i,j}^n}{\Delta x^2} + \frac{\epsilon_{i,j+1}^n + \epsilon_{i,j-1}^n - 2\epsilon_{i,j}^n}{\Delta y^2}. \tag{2.14}$$

Representing ϵ_{ij}^n as the Fourier series given by -

$$\epsilon_{ij}^n = \sum_{k_x=-N}^{N} \sum_{k_y=-N}^{N} E_{k_x k_y}^n e^{i\frac{\pi k_x x}{L}} e^{i\frac{\pi k_y y}{L}} \tag{2.15}$$

and noting that Δx and Δy are equal, the following equation can be determined for the explicit method (by taking only the first mode of the Fourier series) -

$$e^{ij\phi_k}\frac{E_k^{n+1}-E_k^n}{\Delta t}=\frac{2}{\Delta x^2}E_k^n(e^{i(j+1)\phi_k}-2e^{ij\phi_k}+e^{i(j-1)\phi_k}) \tag{2.16}$$

where subscripts for x and y have been dropped and ϕ_k is the phase angle of the k^{th} harmonic given by -

$$\phi_k=\frac{\pi k}{L}\Delta x=\frac{\pi k}{N}. \tag{2.17}$$

Further simplification gives us -

$$\frac{E_k^{n+1}-E_k^n}{E_k^n}=2\gamma(e^{i\phi}-2+e^{-i\phi});\quad \gamma=\frac{\Delta t}{\Delta x^2}. \tag{2.18}$$

which further becomes -

$$\frac{E_k^{n+1}}{E_k^n}=1-8\gamma\sin^2\frac{\phi}{2}=G. \tag{2.19}$$

G is defined as the amplification factor for the explicit scheme for the 2D heat equation and its magnitude must be less than 1 for stability for all ϕ i.e.

$$\left|1-8\gamma\sin^2\frac{\phi}{2}\right|\leq 1 \tag{2.20}$$

which gives the two conditions stability -

$$\begin{aligned}\gamma&\geq 0\\ \gamma&\leq\frac{1}{4}.\end{aligned} \tag{2.21}$$

The explicit method is thus conditionally stable.

2.2.2 Implicit method

Our implicit modified differential equation is given by Eq. 2.3. Using a similar procedure of decomposition we can obtain a evolution equation for the error given by -

$$e^{ij\phi_k}\frac{E_k^{n+1}-E_k^n}{\Delta t}=\frac{2}{\Delta x^2}E_k^{n+1}(e^{i(j+1)\phi_k}-2e^{ij\phi_k}+e^{i(j-1)\phi_k}). \tag{2.22}$$

Reorganization of terms gives us -

$$\frac{E_k^{n+1}}{E_k^n}=\frac{1}{1+8\gamma\sin^2\frac{\phi}{2}}=G. \tag{2.23}$$

Since $|G| \leq 1$ for all ϕ we get -

$$\left| \frac{1}{1 + 8\gamma \sin^2 \frac{\phi}{2}} \right| \leq 1 \tag{2.24}$$

which gives us two conditions for stability -

$$\begin{aligned} \gamma &\geq 0 \\ \gamma &\geq -\frac{1}{4}, \end{aligned} \tag{2.25}$$

which implies that the implicit method for the 2-D heat equation is A-stable since γ is always greater than or equal to zero.

3 Numerical experiments and analysis

3.1 Amplification Factor (G)

The amplification factor expressions derived in the previous section were used to plot the values of G for varying phase angle ϕ_k and stability factor γ. Function definitions and data export were done in Mathematica and plots were made in OriginLab. These are shown in Figs. 3.1-3.3.

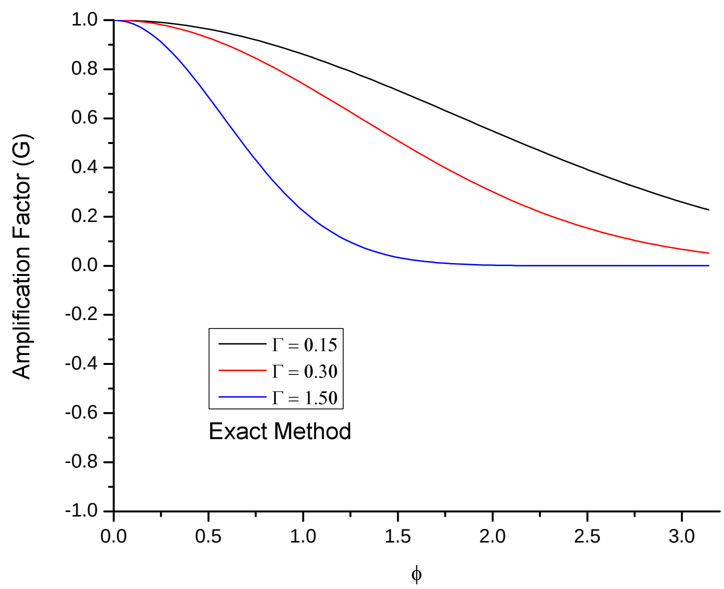

Figure 3.1: Amplification factor of the exact solution of the PDE given by $G_k^e = e^{-\gamma\phi_k^2}$

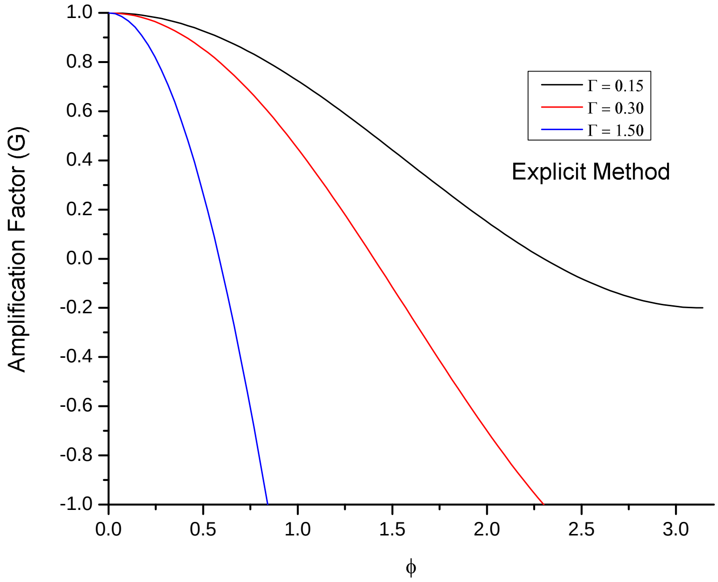

Figure 3.2: Amplification factor of the explicit method to solve the 2-D heat equation

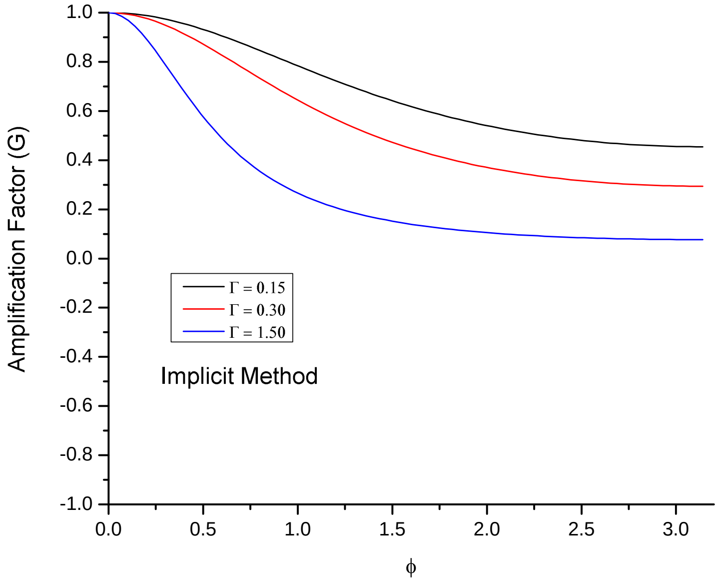

Figure 3.3: Amplification factor of the implicit method to solve the 2-D heat equation

3.2 Iso-contour plots at $t = 0.2s$

The iso-contour of the error between the exact and the explicit methods are shown in Fig. (3.4) for $\gamma = 0.15$ and 0.2 for a 100X100 mesh. It is apparent for the explicit scheme that increasing timesteps cause more error. On careful examination, one can find more zones with blue in the case of $\gamma = 0.2$.

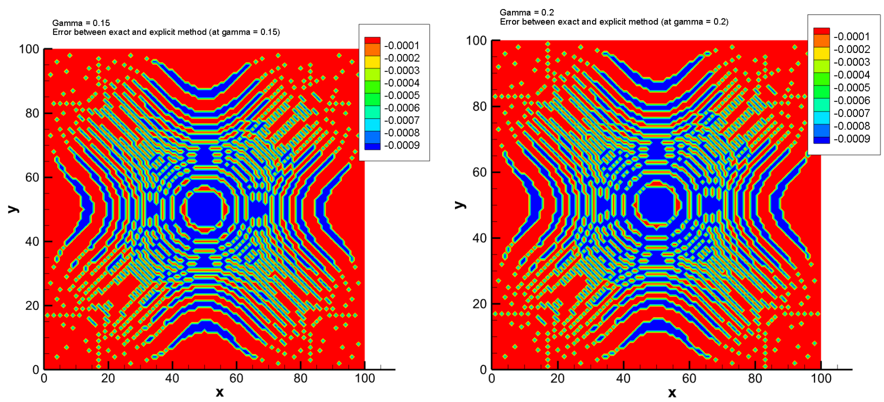

Figure 3.4: Iso-contours of the error between explicit and exact solution (at $t = 0.2s$) - Higher γ causes more error

The iso-contour plots for the steady state solution for both the implicit and explicit method are shown in Figs. (3.5 to 3.7) for the 20X20, 50X50 and 100X100 mesh at $\gamma = 0.15$. The accuracy of both methods is thus ascertained due to the same steady state behavior.

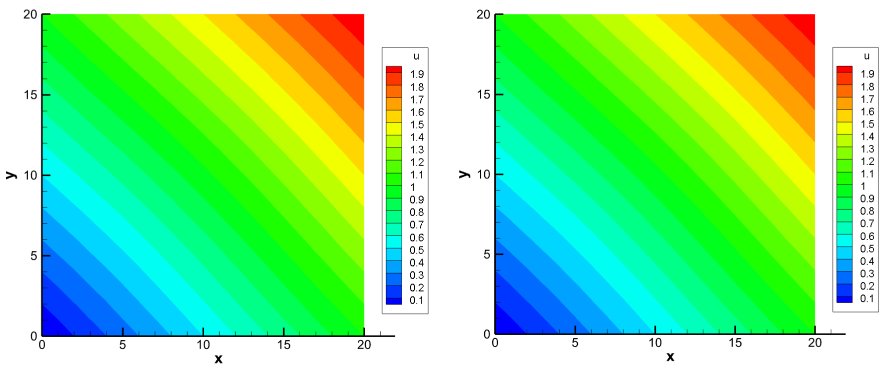

Figure 3.5: Iso-contours of the steady state solution for both the explicit and implicit schemes at $\gamma = 0.15$ and $t = 0.2s$ for a 20X20 mesh

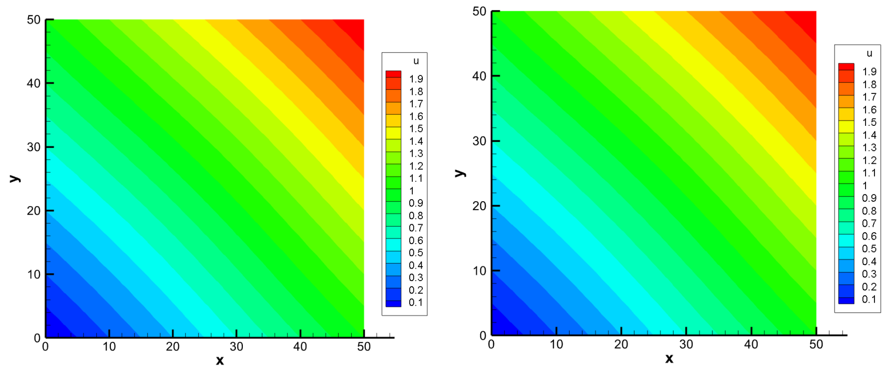

Figure 3.6: Iso-contours of the steady state solution for both the explicit and implicit schemes at $\gamma = 0.15$ and $t = 0.2s$ for a 50X50 mesh

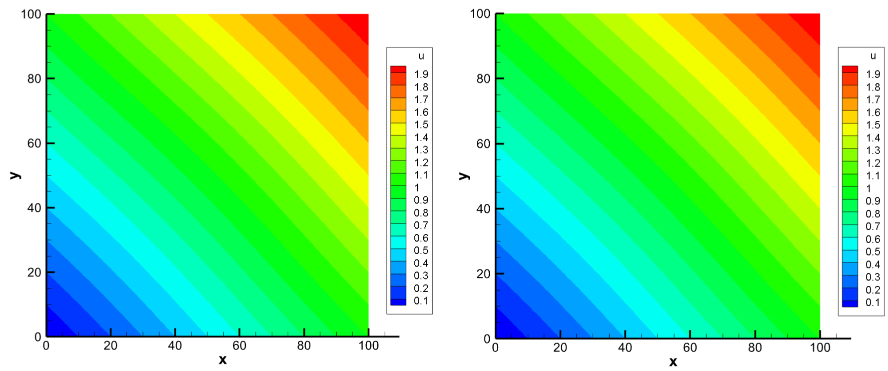

Figure 3.7: Iso-contours of the steady state solution for both the explicit and implicit schemes at $\gamma = 0.15$ and $t = 0.2s$ for a 100X100 mesh

3.3 Impact of time step for 100X100 resolution at $t = 0.1s$

It would seem from the resulting data that at 100X100 resolution, the timestep variation does not cause a sufficiently large increase in error. Ofcourse stability issues dictate that $\gamma \leq 0.25$ for the explicit method. It is seen that the code blows up for anything higher than that.

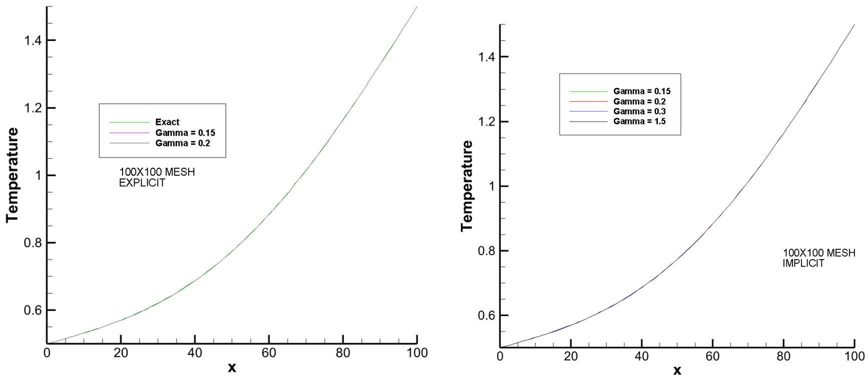

Figure 3.8: Line plots for the effect of timestep variation for the 100X100 mesh. The sufficiently fine resolution precludes any apparent error due to fluctuating γ.

3.4 Impact of spatial resolution for $\gamma = 0.15$ & $t = 0.1s$

It is seen that an increase in spatial resolution corresponds to higher accuracy. When increased, the line plot at $y = 0.5$ tends towards the exact solution for both the implicit and explicit schemes

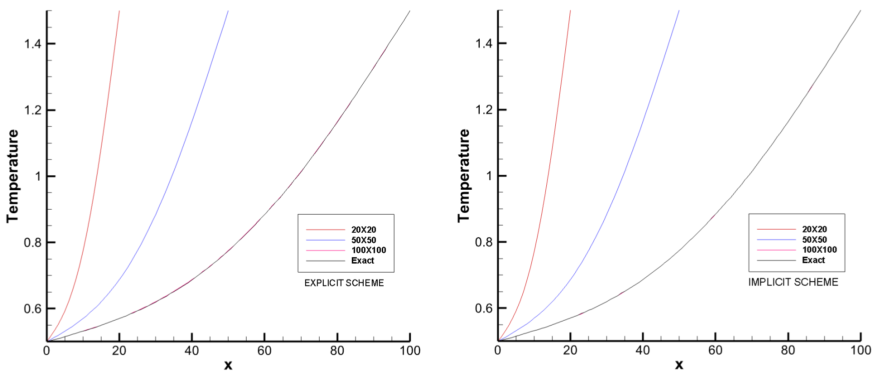

Figure 3.9: Line plots for the effect of resolution variation for $\gamma = 0.15$. Higher resolutions converge to the exact solution

3.5 Impact of numerical method for $\gamma = 0.15$ and 100X100 resolution

It is seen that both numerical methods at 100X100 resolution converge to effectively the same result for varying times.

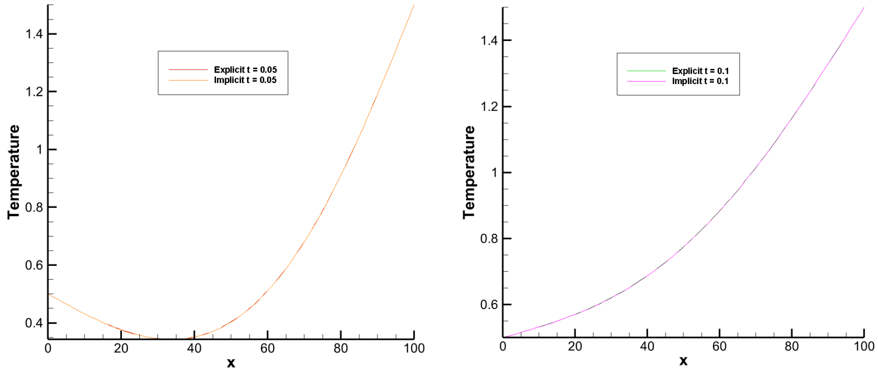

Figure 3.10: Line plots for the effect of numerical method for $\gamma = 0.15$. The 100X100 resolution can be considered to be the exact solution for both schemes

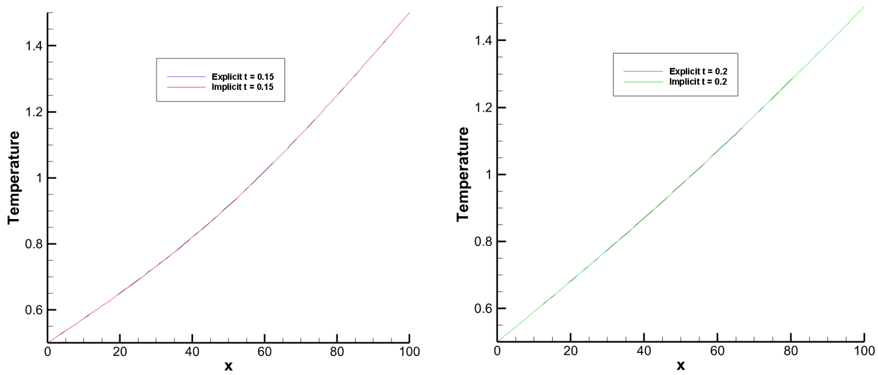

Figure 3.11: Line plots for the effect of numerical method for $\gamma = 0.15$. The 100X100 resolution can be considered to be the exact solution for both schemes

3.6 Comments on computational expense

The following table contains the results for cpu time (in seconds) required for the implicit and explicit schemes for different γ and mesh resolutions. All simulations were carried out using the Gfortran 64 bit compiler on the Cowboy cluster. It is clear that the explicit method is considerably quicker than the implicit scheme provided stability requirements are met. However, the implicit method lets us take larger time steps if needed to reach steady state quickly in a lower number of computations.

Method ($\gamma = 0.15$)	20X20	50X50	100X100
Explicit	0.047	0.296	3.0108
Implicit	2.23	546.84	35416.82